Fachhochschule Köln

University of Applied Sciences Cologne

Abteilung Gummersbach

Fachhochschule Dortmund

Fachbereich Informatik

Verbundstudium Wirtschaftsinformatik

Diplomarbeit

(Drei-Monats-Arbeit)

zur Erlangung

des Diplomgrades

Diplom-Informatiker (FH)

in der Fachrichtung Informatik

„Entwicklung einer Suchmaschine unter Verwendung von Oracle 9*i*AS Portal und Oracle interMedia Text“

Erstprüfer:	Prof. Dr. H. Faeskorn-Woyke
Zweitprüfer:	Prof. Dr. A. Liening
vorgelegt:	Dezember 2002
von cand.	José Matas Nobis
aus	Hans-Böckler-Allee 3
	53177 Bonn
Tel.:	+49 (0)228 - 631430
E-Mail:	Jose.Matas@epost.de
Matr. Nr.:	7 030 537

Inhaltsverzeichnis

Abbildungsverzeichnis V

Tabellenverzeichnis VII

Abkürzungsverzeichnis 8

1 Einleitung 10

1.1 Beschreibung der Thematik 10

1.2 Zielsetzung der Arbeit 11

2 Installation 12

2.1 Rechnerkonfiguration 12

2.2 Pre-Installation 12

2.2.1 Einrichten der Umgebungvariablen 13

2.2.2 Hostname-Konfiguration 13

2.2.3 Download des 8.1.7.3 Patches 13

2.3 Installation von Oracle8*i* Enterprise Edition 14

2.3.1 Laden der Installationsdateien 14

2.3.2 Oracle–Dienste beenden 17

2.3.3 Patch-Installation in das Oracle-Standardverzeichnis 18

2.3.4 Erstellen der Datenbank für Oracle9*i*AS Portal 19

2.4 Installation des Oracle9i Application Servers 25

2.4.1 Beenden der Oracle-Dienste 29

2.4.2 Patch-Installation in das OraHome9*i*AS Standardverzeichnis 29

2.4.3 Neustart der Oracle-Dienste 30

2.4.4 Testen der Installation 30

2.5 Multiple Language Support 31

2.5.1 Installation einer neuen Sprache 31

2.5.2 Auswählen der Sprache 31

3 Oracle Portal 33

3.1 Die Architektur des Portals 33

3.2 Portlets und Pages 35

3.2.1 Portlets 35

3.2.2 Pages 36

3.3 Content Areas 36

3.4 Kategorien 37

4 Implementierung einer Spinne 38

4.1 Programmimplementierung 38

4.1.1 Die Klasse Spider 40

4.1.2 Die Klasse SeiteParsen 41

4.1.3 Die Klasse Threadbegrenzer 48

4.1.4 Die Klasse HTTP 50

4.2 JDeveloper 51

4.3 Programmaufruf 51

5 Intermedia Text 53

5.1 Die Oracle Text Architektur 53

5.2 Datastore 54

5.3 Filter 55

5.4 Sectioner 55

5.5 Lexer 55

5.6 Indexing Engine 56

6 PL/SQL Funktionen 57

6.1 SQL*Plus und PL/SQL 57

6.2 Das PL/SQL wwsbr_api-API Package 57

6.2.1 Funktion Add_content_area 57

6.2.2 Funktion Add_item 59

6.2.3 Funktion Add_folder 63

6.3 Die submit_search-Methode im wwsbr_search_api-Package 63

6.4 Einfügen der URL-Items in die Content-Area mit der LOADER.SQL–Datei. 64

6.5 Laden der Objekte mittels iSQL*Plus 67

7 Suchfunktion mit InterMedia Text 68

7.1 Beispiel für das Suchen in XML–Dokumenten 68

7.2 Der CTXSYS-Benutzer und die CTXAPP-Rolle 69

7.3 Indexierung über das Oracle Portal 70

7.3.1 Gateway–Einstellungen 70

7.3.2 Indexerstellung 71

7.4 Index–Aktualisierungen 74

7.4.1 Automatische Aktualiserung (Synchronisation) 75

7.4.2 Manuelle Aktualisierung 76

7.5 Suchfunktion 76

7.5.1 Basic Search 77

7.5.2 Advanced Search 77

7.5.3 interMedia Search 79

7.6 Operatoren und erweiterte Suchfunktionen 80

7.6.1 Wildcards 80

7.6.2 CONTAINS ALL und CONTAINS ANY 81

7.6.3 Soundex 81

7.6.4 Stem 81

7.6.5 Fuzzy 82

8 Abschluss 83

8.1 Zusammenfassung 83

8.2 Fazit 83

8.3 Ausblick 84

Glossar 85

URL– und Literaturverzeichnis 86

Erklärung 90

Abbildungsverzeichnis

Abb.1 Oracle Universal Installer: Angabe der Installationsverzeichnisse 15
Abb.2 Deselektierung der Komponente Oracle HTTP-Server 1.3.12.0.1a 16
Abb.3 Anzeige der Windows-Dienste 17
Abb.4 Ändern der Eigenschaften für nicht benötigte Dienste 18
Abb.5 Patch Installation in OraHome81 19
Abb.6 Erstellen der Backend - Datenbank 20
Abb.7 Auswahl des Datenbankmodus 21
Abb.8 Wahl der Datenbankoptionen 21
Abb.9 Wahl der Datenbank SID 22
Abb.10 Datenbank-Zeichensatz - Auswahl 22
Abb.11 Ändern der Tablespace- Größe 23
Abb.12 Fortschritt beim Ertellen der Datenbank 24
Abb.13 Meldung des Datenbank-Konfigurationsassistenten 25
Abb.14 Installation von 9*i*AS 26
Abb.15 Editieren des Database Access Descriptors 27
Abb.16 Konfigurationswerkzeuge 28
Abb.17 Informationen nach der Portal-Installation 29
Abb.18 Portal-login 30
Abb.19 3-Ebenen-Architektur (3-Tier) 32
Abb.20 Architektur des Oracle Portals 35
Abb.21 Einrichten einer neuen CONTENT AREA 36
Abb.22 Spider: UML - Diagramm 40
Abb.23 Oracle Text Architektur 53

Abb.24 Auswirkung von p_hide_in_browse = 1 66

Abb.25 Auswirkung von p_hide_in_browse = 0 67

Abb.26 Grant der ctx-Rechte für PORTAL30 70

Abb.27 Connection Pool Parameter 71

Abb.28 Oracle Portal Home Page 73

Abb.29 Indexeigenschaften 73

Abb.30 Suchmaske für die BASIC-Search Funktion 77

Abb.31 Darstellung der Suchergebnisse 78

Abb.32 Advanced Search Suchmaske 79

Abb.33 Suchmaske für die interMedia Text Suche 80

Tabellenverzeichnis

Tab.1 LANGINST-Parameter 31

Tab.2 URL_DATASTORE-Attribute 54

Tab.3 ADD_CONTENT_AREA-Parameter 58

Tab.4 ADD_ITEM-Parameter 60

Tab.5 InterMedia Indizes 74

Abkürzungsverzeichnis

DBA ..... Database Administrator

DOC ..... Dateierweiterung: Document (MS-Word Dateiformat)

FTP..... File Transfer Protocol

GIF ..... Graphical Interchange Format

HREF ..... Hypertext reference keyword

HREF ..... Hypertext Reference Keyword

HTTP..... Hypertext Markup Language

iAS ..... Internet Application Server

IDE..... Integratet Development Enviroment

IR..... Information Retrieval

JPEG..... Dateierweiterung: Joint Photographics Experts Group

MDB..... Dateierweiterung: Microsoft Access Database

OTN ..... Oracle Technology Network

PDF ..... Dateierweiterung: Portable Document Format (Acrobat Reader)

PPT..... Dateierweiterung: Microsoft PowerPoint-Präsentation

SID ..... System Identifier

SQL ..... Structured Query Language

URL..... Uniform Resource Locator

UTF ..... Unicode Transformation Format

WMF ..... Dateierweiterung: Windows Meta File

WWW ..... World Wide Web

XLS ..... Dateierweiterung: Excel Spreadsheet

XML..... Extended Markup Language

ZIP..... Dateierweiterung: Zipped (komprimierte Datei)

1 Einleitung

1.1 Beschreibung der Thematik

Für effizient geführte Unternehmen ist ein schnelles Zugreifen auf eigene Informationen über Intra- und Internet ohne Frage überlebenswichtig. Viele Firmen haben mit großem Aufwand Text–Dokumente innerhalb von Datenbanken strukturiert abgelegt. Diese Dokumente sind aber oft nur ein kleiner Bruchteil der im Betrieb anfallenden Dokumente. Internet-Seiten, E-mails, Faxe und viele andere Dokumente bleiben außen vor. Teilweise sind die in Datenbanken abgelegte Dokumente auch statisch, sie werden nicht oder nur aufwendig aktualisiert. Mitarbeiter und Öffentlichkeit müssen aber raschen Zugriff auf aktuelle und für Sie abgestimmte Informationen haben. Mitarbeiter sollten die Möglichkeit haben, Dokumente zu bearbeiten und eigene Dokumente für andere Mitarbeiter zur Verfügung zu stellen. Zudem sollte die Öffentlichkeit speziell für Sie aufbereitete Informationen über das Internet erhalten können. Der Zugriff auf unterschiedliche dynamische Dokumente geschieht idealerweise über einen gemeinsamen Gateway und ist plattform-unabhängig. Diese Bereitstellung dynamischer Informationen stellt für den Betrieb einen großen Kostenvorteil dar.

Ein Werkzeug, das diesen Ansatz verfolgt ist Oracle Portal. Alle Dokumente eines Betriebes werden in einer Oracle-Datenbank browserbasiert verwaltet. Es wird den Mitarbeitern eine einheitliche Oberfläche zur Verfügung gestellt über die sie weltweiten Zugriff auf die Dokumente haben. Eine produktive globale Verflechtung der Informationen wird möglich.

Die Unternehmensdaten werden zusammen mit allen anfallenden Dokumenten, WWW–Seiten, Office- und PDF- (soweit nicht aus Bitmaps bestehend) Dateien, etc. in einer Oracle Datenbank gespeichert. Die Verwaltung aller Daten erfolgt zentral wodurch die Wartung der Daten erleichtert wird. Ein Backup der Datenbank beinhaltet alle Unternehmensdaten zusammen mit deren benutzerspezifischen Informationen.

In diesem Zusammenhang werden zwangsläufig sehr große Mengen an heterogenen Daten angesammelt und es entsteht schnell ein Bedarf, diese Dokumente gezielt zu durchsuchen um Informationen rasch bereitzustellen.

Die Suchfunktion interMedia Text von Oracle erlaubt komplexe Suchanfragen innerhalb solche großen heterogene Dokumentenbestände.

1.2 Zielsetzung der Arbeit

Im Anschluß an die Projektarbeit „Klassifizierung und theoretische Grundlagen von Suchmaschinen“ soll eine Suchmaschine unter Verwendung von Oracle Portal 3.0 entwickelt werden. Die Suchfunktion soll die verschiedenen Text–Dokumente der Website http://www.gm.fh-koeln.de/~faeskorn/[1] vollständig indexieren. Diese Domäne beinhaltet zur Zeit über 11000 Dateien und besteht aus text / HTML , ZIP, PDF, XLS, DOC, MDB, PPT sowie (hier nicht weiter relevante) GIF, WMF und JPEG Dateien.

Als Website (oder Site) wird hier die Gesamtheit der Dokumente verstanden, die sich im Internet unter o.g. Domäne befinden und mit der Startseite direkt oder indirekt verlinkt sind.

[1] Es sollte durch Analogie möglich sein, eine Suchmaschine für eine beliebige Website zu implementieren.

2 Installation

Installiert wurden Oracle 8.1.7.0.0 Enterprise Edition und Oracle 9iAS 1.0.2.2.2. Beide Datenbanken wurden auf 8.1.7.3.0 gepatcht. Die installierte Portal Versionsnummer lautet entsprechend 3.0.9.8.0.

Da jede Rechnerkonfiguration anders ist, kann nicht davon ausgegangen werden, dass Installationsanweisungen eins zu eins übernommen werden können. Die Installation der Oracle Produkte verläuft sehr hackelig. Die Metalink–Seiten[2] von Oracle bieten Hilfestellungen bei verschiedenen Problemen.

Da die Installation ein sehr zeitraubender Prozeß sein kann, werden hier einige wichtige Aspekte des Installationsprozesses ausführlicher beschrieben.

Eine sehr detaillierte Installationsanleitung kann von Oracle unter http://portalstudio.oracle.com/pls/ops/docs/FOLDER/COMMUNITY/OTN_CONTENT/DOCUMENTATION/ORACLE9IASPORTALINSTALLGUIDE.ZIP heruntergeladen werden.

2.1 Rechnerkonfiguration

Für diese Arbeit wurde ein einzelner Rechner mit Windows 2000 (5.00.2195 Service Pack 2) als Betriebssystem benutzt. Die von Oracle benötigten Ressourcen sind aber sehr groß, so dass idealerweise ein für die 3-Ebenen–Architektur (Kap. 3.1) optimiertes System benutzt werden sollte. Dabei sollte der Datenbankserver über schnelle Platten und einen großen Hauptspeicher und der Applikationsserver über viel Hauptspeicher und einen schnellen Prozessor verfügen.

2.2 Pre-Installation

Vor der Installation sollten die Installationsanweisungen des Produktes sorgfältig durchgelesen werden. Es ist an dieser Stelle empfehlenswert die aktuellen Systemeinstellungen bzw. die Registry von Windows zu sichern. Sollte die Oracle-

[2] [METALINK]

Installation scheitern, ist ansonsten u.U. ein manueller Eingriff in die Registry nötig um das Produkt nochmal installieren zu können.

2.2.1 Einrichten der Umgebungvariablen

Für die Installation ist die Anmeldung am Betriebssystem mit Administrator-Rechte erforderlich.
Es wird eine Systemvariable NLS_LANG mit dem Wert .UTF8 angelegt. Der vorangehende Punkt beim Wert .UTF8 darf nicht ignoriert werden da sonst später die Portal–Installation ohne Meldung abgebrochen wird. Für die TEMP und TMP Variablen sollte eine Partition mit genügend Platz ausgewählt werden.

2.2.2 Hostname-Konfiguration

Der Computername muss in Kleinbuchstaben geschrieben werden (in diesem Fall: *home*), weil das Installationsprogramm später diese Information ausliest und die Konfigurationsdateien damit erstellt. Die Netwerkkennung muß, falls vorhanden, die komplette Domänen-Erweiterung beinhalten.

2.2.3 Download des 8.1.7.3 Patches

Von der Metalink–Seite[3] muss der Patch mit der Nummer 2189751 (8.1.7.2.3) runtergeladen werden. Dieser Patch ist kummulativ, d.h. die Patchsets für Version 8.1.7.1.1 (Nummer 1711240) und Version 8.1.7.2.1 (Nummer 1882450) müssen nicht gesondert installiert werden. Alle 3 Patches beinhalten Upgrades, die InterMedia Text berühren und mussten im Laufe der Enstehung dieser Arbeit installiert werden. Ohne das 8.1.7.1.1–Update ist durch eine Instabilität des Listeners die Portal-Installation gar nicht möglich. Ohne das 8.1.7.2.3– pdate lässt sich aus dem Portal heraus kein InterMedia–Index erstellen.

Alternativ zum letzten Patchset, aber nur bedingt zu empfehlen, gibt es bei http://metalink.oracle.com/metalink/plsql/showdoc?db=NOT&id=158368.1 eine

[3] [METALINK]

Anleitung zur Umgehung des Problems. Oracle empfiehlt, ständig die letzten Produktversionen zu benutzen.

2.3 Installation von Oracle8*i* Enterprise Edition

Die Installation des Oracle 9iAS (Release 1) Portals mit der Oracle 9i Datenbank als Backend ist laut Angaben von Oracle nicht ohne weiteres möglich, da 9iAS noch auf der Oracle 8.1.7 Datenbank basiert. Empfohlen wird die Installation des Oracle 9iAS-Portals mit Oracle 8.1.7. Beide Produkte müssen in getrennten Standardverzeichnissen (homes) installiert werden.

2.3.1 Laden der Installationsdateien

Die Installationsdateien der Oracle8*i* (8.1.7.0.0) Enterprise Edition–CD können temporär auf die Festplatte kopiert werden, um den Installationsprozeß zu beschleunigen. Mit SETUP.EXE wird der Oracle Universal Installer gestartet. Dieser stellt eine einheitliche Oberfläche zur Installation der verschiedenen Oracle–Produkte dar.

Im ersten Dialogfenster (Abb.1) werden die Quell- und Zielverzeichnisse der Installation festgelegt sowie der Name des Oracle Home–Verzeichnisses (hier OraHome81).

Abb. 1: Oracle Universal Installer: Angabe der Installationsverzeichnisse.

Weiter wird die Installation von Oracle8i Enterprise Edition 8.1.7.0.0 gewählt und bestätigt. Die Installation sollte mit der Option BENUTZERDEFINIERT ausgeführt werden. Da das Installationsprogramm eigens in Java inplementiert wurde und keinen Gebrauch von den Windows-Standard-Elementen macht, sollte man sich von der unheitlichen teils zweischprachigen Menüführung nicht irritieren lassen.

Die Komponente „Oracle HTTP-Server 1.3.12.0.1a“ wird deselektiert (Abb. 2). Der auf Apache basierende HTTP-Server wird später bei der Oracle 9*i*AS-Installation installiert und eingerichtet.
Es wird ein Pfad für die Komponenten, die nicht im Oracle Standardverzeichnis gespeichert werden, selektiert und bestätigt.
Die Berechtigungsprüfungsmethodenauswahl wird ohne Auswahl der angebotenen Methoden mit WEITER bestätigt.

Abb. 2: Deselektierung der Komponente Oracle HTTP-Server 1.3.12.0.1a.

Die „Datenbank erstellen“-Abfrage wird mit „NO“ markiert und bestätigt.

Es wird ein Überblick der zu installierenden Produkte eingeblendet. Die Installation kann nun mit Betätigung der Schaltfläche „INSTALLIEREN“ gestartet werden.

Das Installationsprogramm startet mit der Anzeige eines Balkens, der den Fortschritt der Installation anzeigt und mit der Angabe der Pfadinformationen der Protokolldateien für die Installation. Sollte die Installation nicht erfolgreich sein, kann in diesen Dateien nach eine Problemlösung gesucht werden indem nach den dort protokollierten Angaben in den Oracle Technology Network[4]–Seiten oder bei Metalink im Internet gesucht wird.

Nach der Installation wird automatisch der Net8–Konfigurationsassistent gestartet. Die Option TYPISCHE KONFIGURATION muss aktiviert werden. Nach erfolgreicher Installation wird dies mit einer entsprechenden Meldung quittiert, mit BEENDEN wird das Installationsprogramm verlassen.

[4] [OTN]

2.3.2 Oracle–Dienste beenden

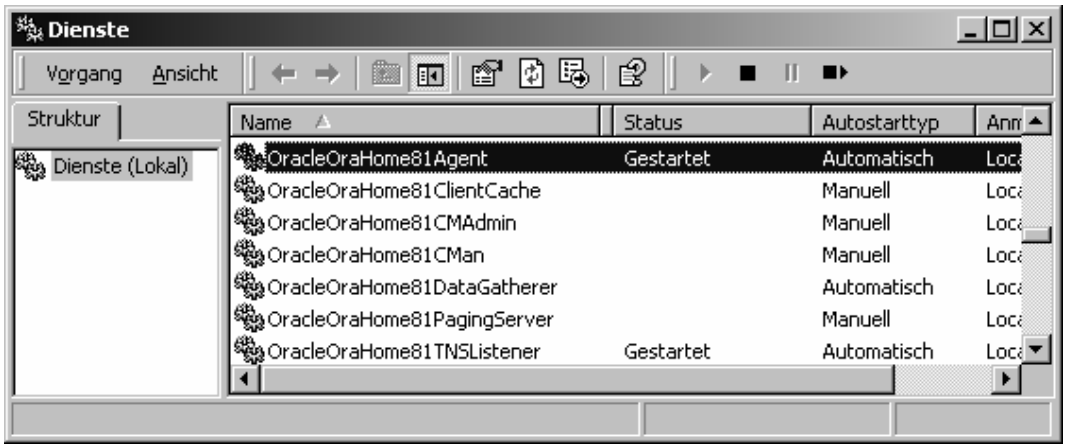

Abb. 3: Anzeige der Windows - Dienste.

Die Oracle–Dienste müssen vor der Installation des Patches beendet werden. Dienste die nicht benötigt werden, werden außerdem auf Autostarttyp Manuell eingestellt, um zu verhindern, dass sie nach dem nächsten Bootvorgang geladen werden.

Die Oracle – Dienste befinden sich unter *START -> SYSTEMSTEUERUNG* -> Icon *VERWALTUNG* -> Icon *DIENSTE* (Abb. 3). Die Oracle Dienste beginnen alle mit „Oracle“ + Name des Standard- (oder Home) –Verzeichnisses, so dass diese in der Sortierung nach Namen der Dienste-Auflistung gut aufzufinden sind.

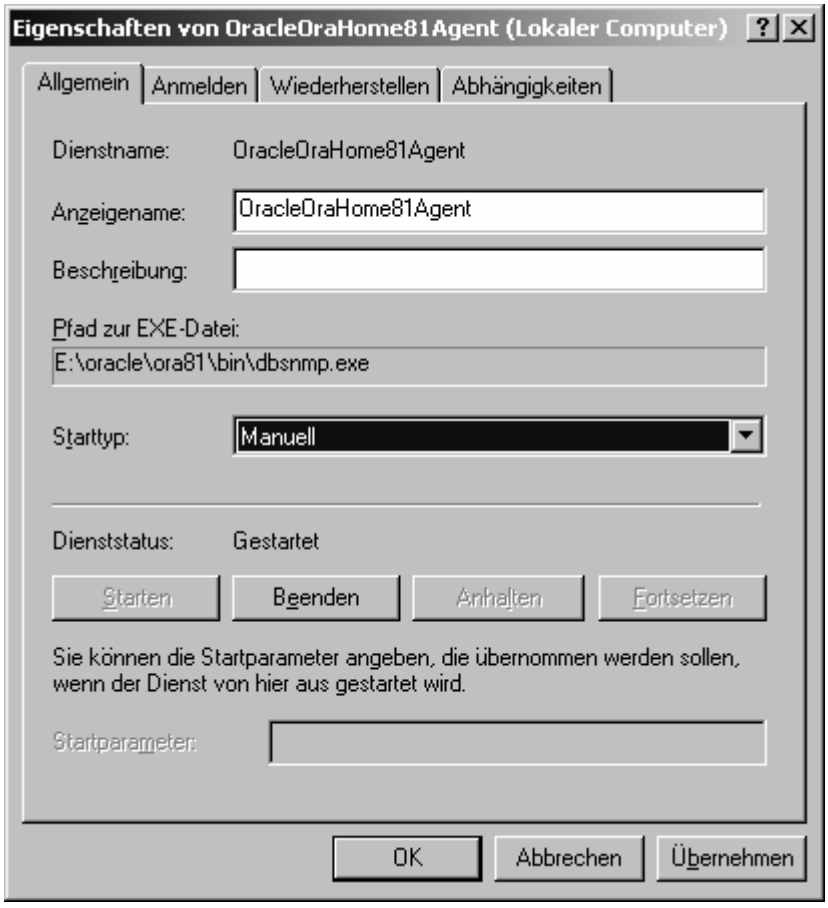

Abb. 4: Ändern der Eigenschaften für nicht benötigte Dienste

Durch einen Doppelclick auf den jeweiligen Dienst wird ein Fenster mit den Eigenschaften des Dienstes geöffnet. Folgende Dienste müssen nun beendet und auf „STARTTYP: MANUELL „ (Abb. 4) gesetzt werden:

- *OracleOraHome81*Agent
- *OracleOraHome81*DataGatherer

Der Dienst *OracleOraHome81*TNSlistener muss beendet werden und auf „STARTTYP: AUTOMATISCH“ gesetzt werden. Somit wird er nach dem nächsten Bootvorgang automatisch geladen.

2.3.3 Patch-Installation in das Oracle-Standardverzeichnis

Die Installation des Patches geschieht ähnlich wie die Hauptinstallation und wird ebenfalls vom Universal Installer geführt. Als Oracle-Standardverzeichnis wird das vorherige Standardverzeichnis OraHome81 gewählt (Abb.5).

Abb. 5: Patch Installation in OraHome81

2.3.4 Erstellen der Datenbank für Oracle9*i*AS Portal

Oracle 9*i*AS benötigt im OraHome81-Verzeichnis eine Backend Datenbank, in der die eigentlichen Daten gespeichert werden. Um diese einzurichten muss der Datenbank-Konfigurationsassistent gestartet werden. Er befindet sich im Startmenü unter Start -> Programme -> Oracle - OraHome81 -> Database Administration -> Database Configuration Assistant.

Die Prozedur ERSTELLEN EINER DATENBANK wird im ersten Fenster gewählt und mit WEITER bestätigt (Abb. 6).

Abb. 6: Erstellen der Backend - Datenbank.

Als Datenbanktyp wird BENUTZERDEFINIERT gewählt. Als primärer Anwendungszweck wird MEHRZWECK aktiviert. Je nach Ressourcen und erwartete gleichzeitige Client–Verbindungen kann zwischen ein dedizierter Modus und ein Multi-Threaded Betrieb des Servers gewählt werden (Abb. 7). Beim dedizierten Server-Modus wird von der Datenbank für jede Client-Verbindung eine dedizierte Ressource zugewiesen, beim gemeinsamen Server-Modus wird von verschiedenen Client-Verbindungen ein gemeinsamer Pool von Ressourcen verwendet. Letzteres ist dann sinnvoll, wenn eine große Anzahl von Benutzern gleichzeitig auf die Datenbank zugreift.

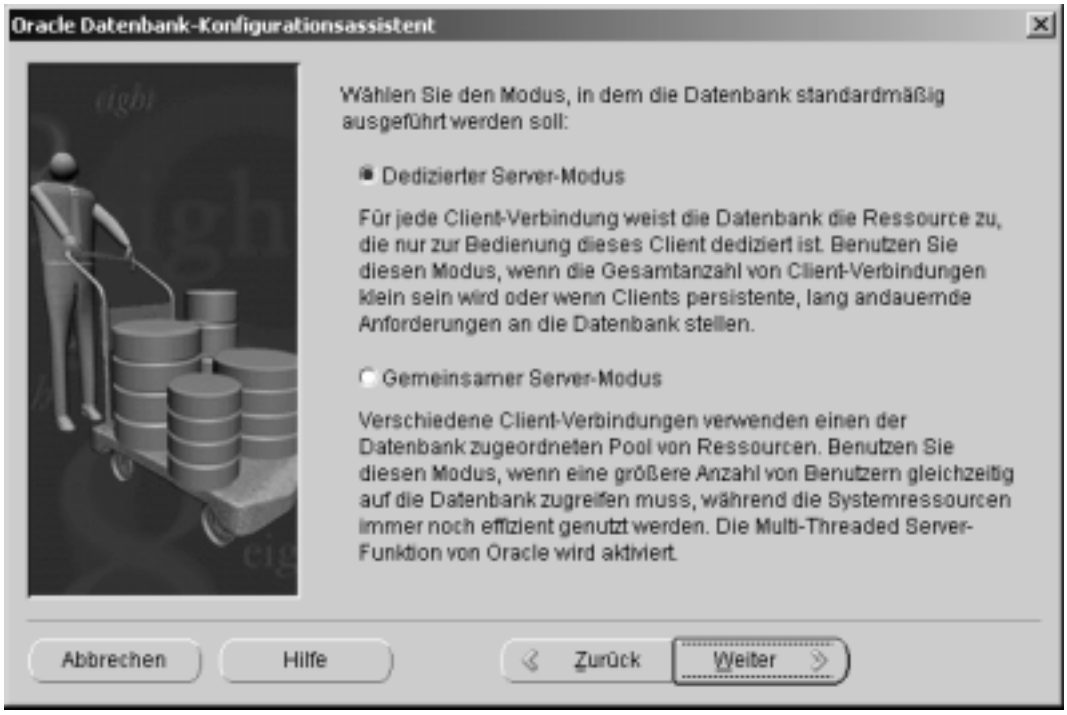

Abb. 7: Auswahl des Datenbankmodus

Oracle Time Series, Oracle Spatial, Oracle Visual Information Retrieval und Advanced Replication werden deselektiert, Oracle 9*i*AS Portal benötigt diese Optionen nicht (Abb. 8).

Abb. 8: Wahl der Datenbankoptionen

Weiter müssen ein globaler Datenbankname und eine SID gewählt werden (hier ORCL). Jede Oracle-Instanz wird mit einer solchen bis zu vier Zeichen langen Zeichenkette einmalig bezeichnet. Die SID wird von den Oracle Komponenten benutzt, um die Anwendungen mit der korrekten Instanz zu verbinden (Abb. 9).

Abb. 9: Wahl der Datenbank SID

Über die Schaltfläche „Zeichensatz ändern“ gelangt man ins Menü für die Datenbank-Zeichensatz-Konfiguration (Abb.10). UTF8 bietet hier mit dem UNICODE[5]-Zeichensatz die meisten Möglichkeiten auch mit fremdsprachigen Dokumenten umzugehen.

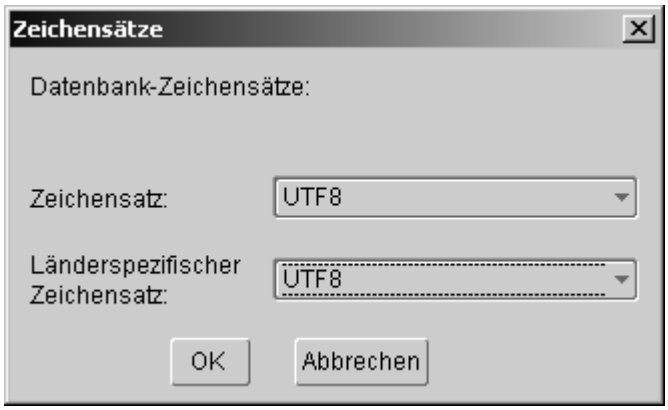

Abb. 10: Datenbank-Zeichensatz - Auswahl

[5] Vgl. [UNICODE]

Die Größe der Tablespaces sollte im Laufe der Konfiguration (Abb. 11) geändert werden. Für jedes Tablespace wird im Installationsfenster eine Registerkarte angezeigt. Als Mindestgrößen sollen folgende Richtwerte gelten:
In der System–Registerkarte wird die Größe auf 600 MB geändert.
In de Werkzeuge–Registerkarte wird die Größe auf 1 MB geändert.
Das Benutzer–Tablespace wird unter der Benutzer–Registerkarte auf 300 MB vergößert.
In der Rollback–Registerkarte wird die Größe auf 50 MB verändert.
Die Größe des Index–Tablespace wird auf 10 MB vergrößert.

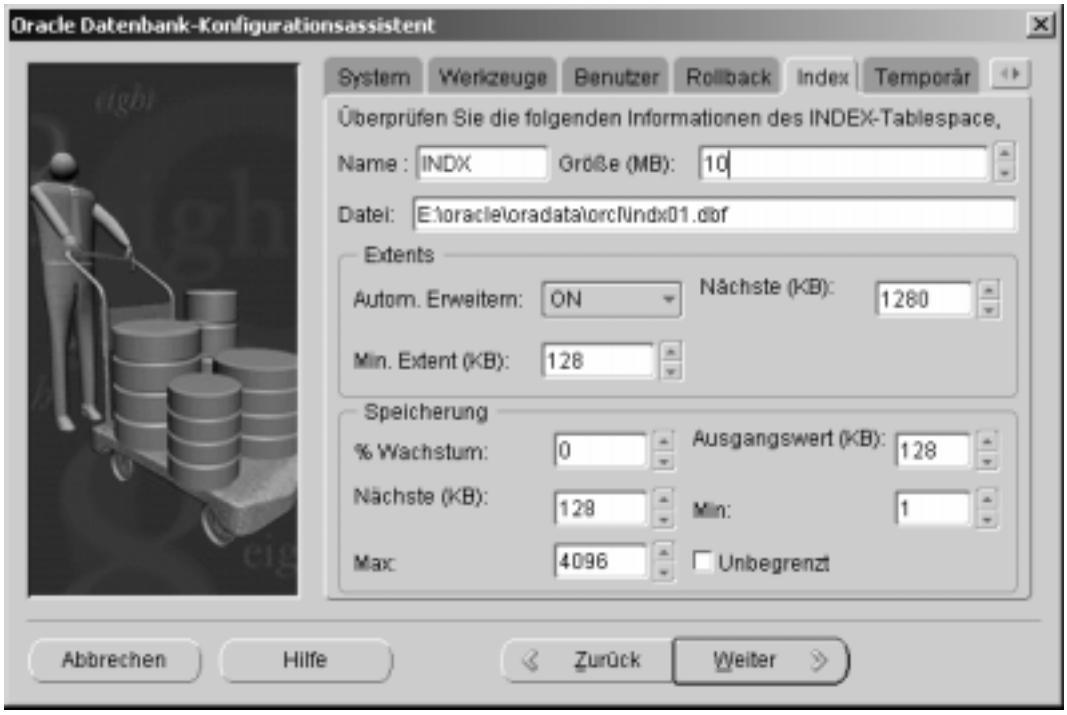

Abb. 11: Ändern der Tablespace- Größe

In der Temporär–Registerkarte wird die Größe auf 20 MB verändert.
In der Intermedia–Registerkarte wird die Größe des Tablespaces auf 20 MB verändert.
Als Richtwerte für die Größe der Redo-Log – Dateien gelten weiter jeweils 25600 KB für die Redo-Log-Dateien 1 , 2 und 3. Alle anderen Werte können unverändert übernommen werden. Je nach Bedarf können die Einstellungen auch zu einem späteren Zeitpunkt im Enterprise-Manager geändert werden.

Die Datenbank kann nun vom Datenbank-Konfigurationsassistenten automatisch erstellt werden. Ein Balken zeigt den Fortschritt der Installation an (Abb.12).

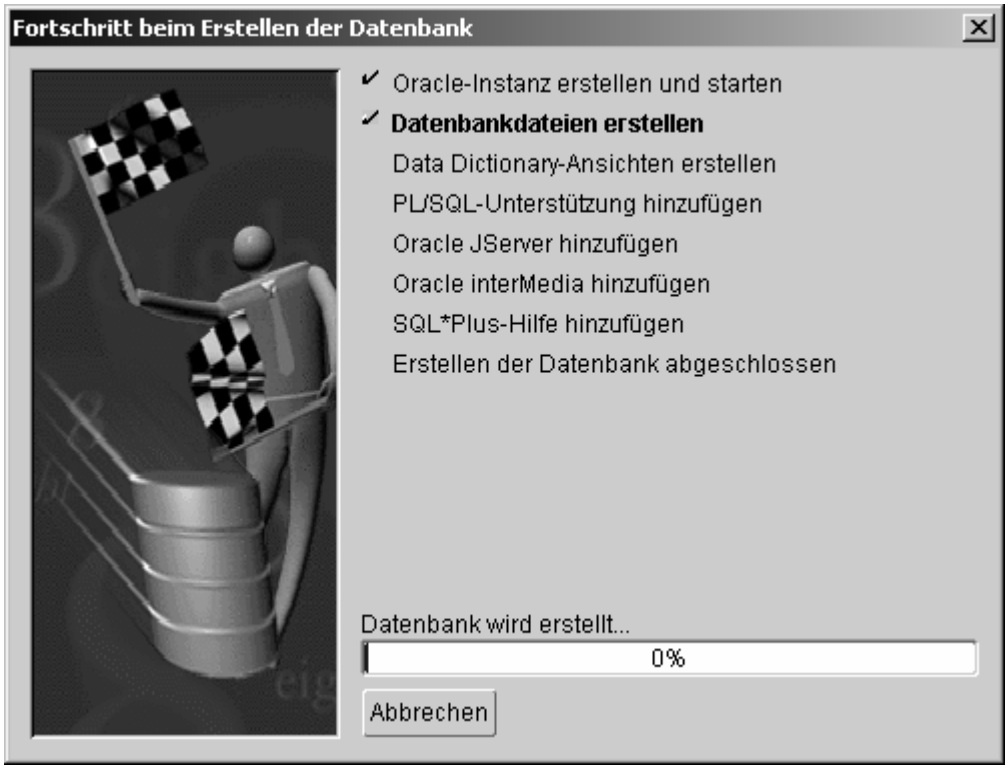

Abb. 12: Fortschritt beim Erstellen der Datenbank

Nach erfolgreicher Erstellung der Datenbank erscheint eine entsprechende Meldung (Abb. 13) in der der Pfad für die Installationslogdatei und die aktuellen Passwörter für die DBA-Accounts (SYS und SYSTEM) angezeigt wird.
An dieser Stelle bietet es sich an, aus Sicherheitsgründen die Kennwörter zu ändern. Nach erfolgreicher Installation muß der Rechner neu gestartet werden.

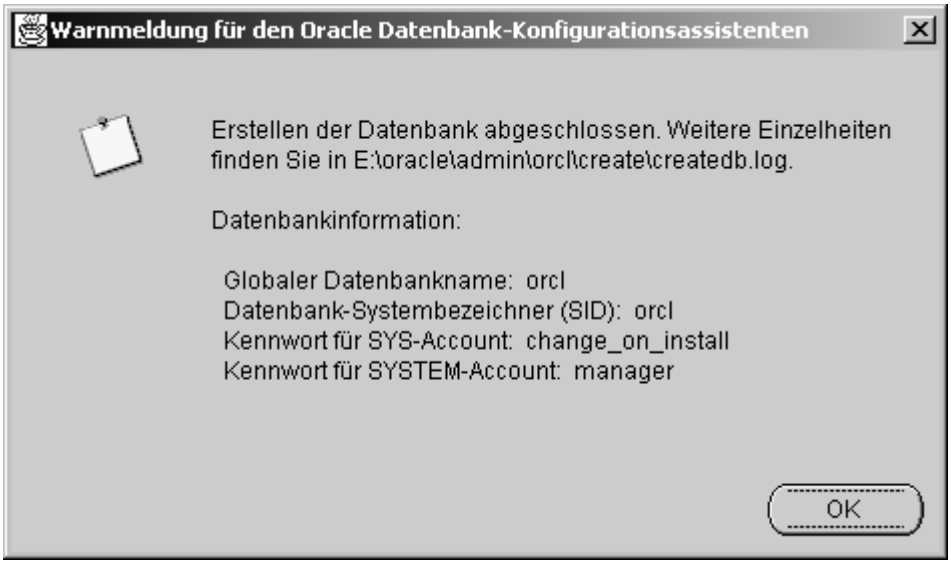

Abb. 13: Meldung des Datenbank-Konfigurationsassistenten

2.4 Installation des Oracle9i Application Servers

Für die Installation ist es ratsam, die drei Installations-CD's in 3 Verzeichnissen Disk1, Disk2 und Disk3 auf die Festplatte zu kopieren. (Alternativ kann die Installation auch von den CD's aus erfolgen, so erspart man sich die Aufforderungen zum Wechseln der Datenträger) Um die Installation zu starten, muss die Datei iSetup.exe (Disk1) ausgeführt werden. Unter Installation Types muss die MINIMAL Edition gewählt und bestätigt werden.

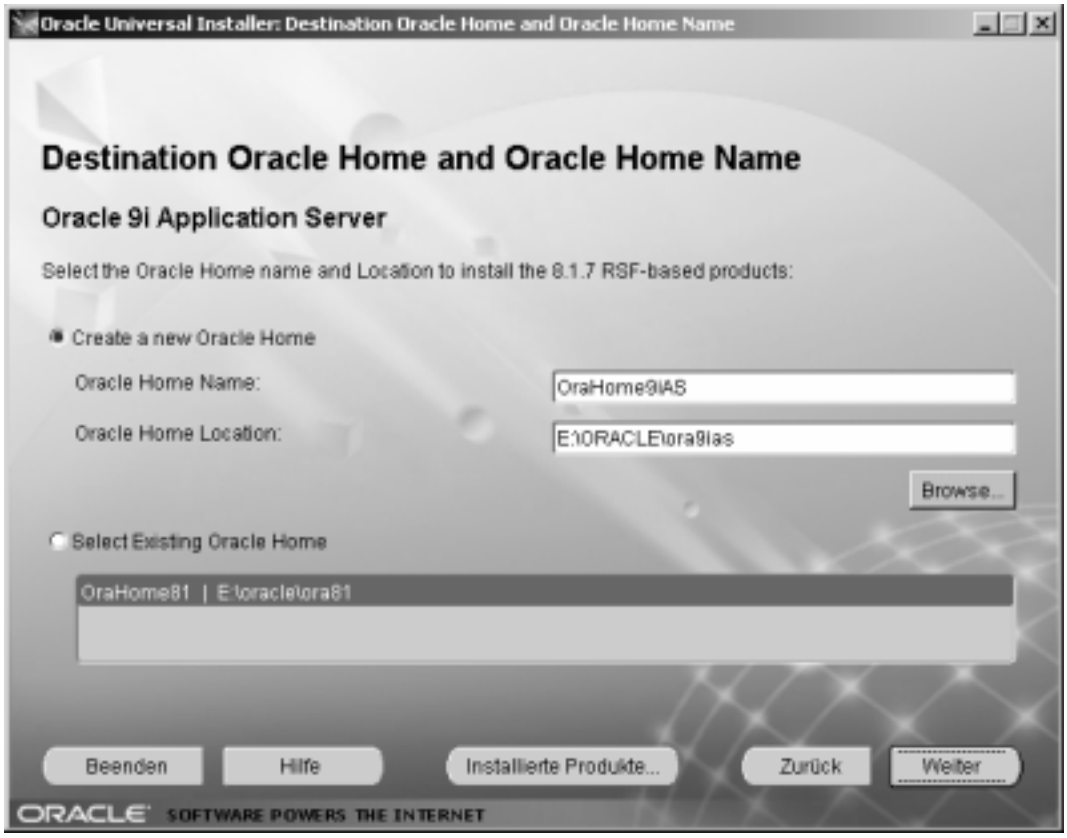

Abb. 14: Installation von 9iAS

Die Installation muss in einen neuen Standardverzeichnis (Home) erfolgen und darf auf keinen Fall in das schon bestehende OraHome81 installiert werden. Hier wurde als Name OraHome9iAS gewählt und als Home Location E:\ORACLE\ora9ias (Abb. 14).
Bei der Konfiguration des Apache-Listeners wird ein Name für den Database Access Descriptor (DAD) eingegeben, Vorgabe ist portal30. Unter Connect String müssen der Hostname, Port und SID der Backend–Datenbank, getrennt durch Doppelpunkte eingegeben werden. (Hier home:1521:orcl)
Als DAD–Name für den Login–Server wurde die Vorgabe portal30_so übernommen. Nach erfolgreicher Installation können weitere DAD's unter http://<hostname>:port/pls/admin_/gateway.htm (Abb.15) erzeugt werden.

Abb. 15: Editieren des Database Access Descriptors

Unter „Wireless Edition repository information" müssen weiter der Hostname, die Port–Nummer des Net8 Listeners und die SID der Datenbank eingegeben werden. Als Username / Passwort können wireless / wireless eingegeben werden, als System-Passwort „manager". Vor der Installation wird noch ein Überblick über die zu installierende Produkte angezeigt. Während der Installation werden erneut die Pfadangaben für die Installationslogdateien angezeigt.

Nach erfolgreicher Installation wird automatisch der Assistent für die Net8-Konfiguration gestartet. Die angebotene typische Konfiguration kann übernommen werden.

Nach dem automatischen Start des HTTP-Servers beginnt die assistentengeführte Portal-Konfiguration (Abb.16).

Abb. 16: Konfigurationswerkzeuge

Als Verbindungsinformation müssen das SYS-Passwort (change_on_install, wenn nicht vorher geändert) und (erneut) den connect-String (hostname:port:sid) der Backend-Datenbank eingegeben werden. Alle weiteren Vorgaben können übernommen werden.

Nach erfolgreicher Installation wird eine Meldung mit den URL's ausgegeben, die für den Zugang zum Portal relevant sind (Abb. 17).

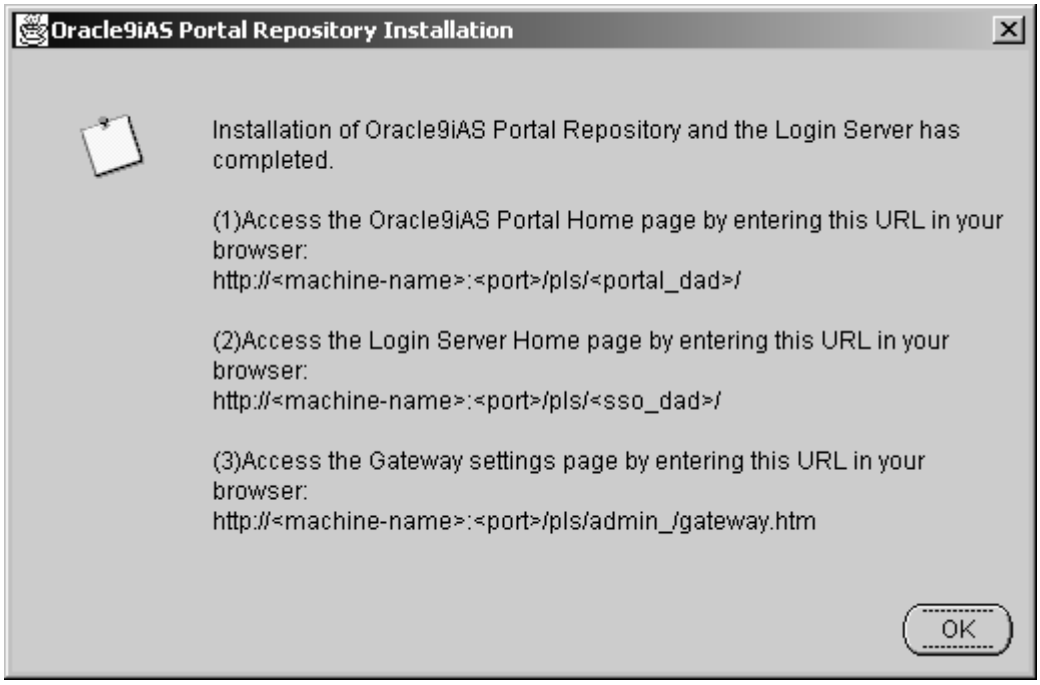

Abb. 17: Informationen nach der Portal-Installation

2.4.1 Beenden der Oracle-Dienste

Als Vorbereitung für die Patch-Installation müssen analog zu der Patch-Installation für das 8.1.7–Home (Kap. 2.3.3) auch hier nach einen Neustart des Rechners alle Oracle – Dienste beendet werden. Dienste die nicht benötigt werden, werden zudem unter EIGENSCHAFTEN auf Starttyp:MANUELL gesetzt. (Alle Oracle-Dienste außer *OracleOraHome81TNSListener*, *OracleOraHome9iASHTTPServer* und *OracleServiceORCL*)

2.4.2 Patch-Installation in das OraHome9*i*AS Standardverzeichnis

Da Oracle 9iAS auf Oracle 8.1.7 basiert, ist auch hier ein Upgrade erforderlich. Die Installation vom 8.1.7.3 Patchset erfolgt analog wie beim OraHome81–Verzeichnis, anstatt OraHome81 wird jetzt nur das OraHome9iAS-Verzeichnis angegeben (auswählen in der Drop-Down Liste).

2.4.3 Neustart der Oracle-Dienste

Folgende Dienste können nun gestartet werden:

- OracleOraHome81TNSListener
- OracleOraHome9iASHTTPServer
- OracleServiceORCL

Ein Neustart des Rechners hat auch ein Starten o.g. Dienste zur Folge, sofern der Starttyp noch auf automatisch eingestellt ist.

Je nachdem welcher Patch als letzter installiert wurde, ändert sich auch das standardmäßige Oracle-Home Verzeichnis. Mit dem Ora–Home Selector–Werkzeug kann das Oracle-Home Verzeichnis ggf. wieder auf Ora81 eingestellt werden. Dies erfordert einen Neustart des Rechners.

2.4.4 Testen der Installation

Im Internet Explorer kann die URL des Rechners gefolgt von /pls/ eingegeben werden (http://localhost/pls/)

Abb. 18: Portal - login

Über das Portal kann man sich nun als *portal30* mit Kennwort *portal30* einloggen (Abb.18).

2.5 Multiple Language Support

Nach der Installation steht im Portal nur die Sprache "us" (amerikanisches English) zur verfügung. Um weitere Sprachen zu aktivieren, in diesem Falle Deutsch, muss wie folgt vorgegangen werden:

2.5.1 Installation einer neuen Sprache

Um eine neue Sprache zu installieren muß die Scriptdatei `langinst.cmd`[6] vom `$ORACLE_HOME\portal30\admin\plsql` -Verzeichnis geladen und ausgeführt werden.
Die Scriptdatei wird mit Parametern folgendermaßen aufgerufen:

```
langinst.cmd [-s portal_schema] [-p portal_password] [-o sso_schema]
[-d sso_password] [-c connect_string] [-l language] [-available]
```

Die Parameter:

`-s portal_schema`	(Default = PORTAL30) Portal – Datenbankobjekt
`-p portal_password`	(Default = PORTAL30) Passwort für das Portal Schema
`-o sso_schema`	(Default = PORTAL30_SSO) Schema für den Login Server
`-d sso_password`	(Default = SSO_PORTAL30) Passwort für den Login Server
`-c connect_string`	Optionaler connect string der Oracle-Instanz.
`-l language`	Kürzel für die zu installierende Sprache (d für Deutsch)
`-available`	Optionaler Parameter. Wenn mit aufgeführt, steht die Sprache den Usern für Übersetzungen zur Verfügung.

Tab. 1: langinst-Parameter

Beispiel:

```
langinst.cmd -s portal30 -p portal30 -o portal30_sso -c
orcl -l d -available
```

2.5.2 Auswählen der Sprache

Die Auswahl der Sprache wird über das “Set Language Portlet” gesteuert.

[6] [Metalink 130328.1]

Nach der Änderung muß die Anzeige der Seiten aktualisiert werden. Damit die angezeigte Sprache beim Aufrufen der Seiten automatisch aktualisiert wird, kann es sinnvoll sein, den PL/SQL–Cache zu deaktivieren. Dies geschieht über die Adresse `http://<hostname>/pls/portal30/admin_/cache.htm`,
indem die “Enable PLSQL Caching”–Option deaktiviert wird.

3 Oracle Portal

Ein Portal ist ein browserbasiertes Entwicklungswerkzeug, in dem Informationen und Anwendungen im Internet oder im Intranet zur Verfügung gestellt werden können. Die über einen Login-Server authentifizierten Benutzer des Portals können eigene Objekte einbinden und das Layout entsprechend den eigenen Bedürfnissen verändern.
Die Administration des Portals geschieht zentral, es ist kein spezieller Client nötig, die Entwicklung und Administration wird über einen Browser (fern)gesteuert.
Die HTML-Seiten und die Daten eines Unternehmens können mit einheitlichen Layouts erstellt und zusammen in einer Datenbank gespeichert werden. Dies erleichtert die Administration, z.B. im Hinblick auf Backups und Recovery-Funktionen.
Eine Internet-Site könnte idealerweise komplett innerhalb des Portals konzipiert und erstellt werden, dies würde auch die Erstellung einer Suchmaschine erleichtern, da eine solche Funktion für Content-Areas des Portals schon fertig implementiert ist.

3.1 Die Architektur des Portals

Oracle Portal ist ein Teil des Oracle internet Application Server (OiAS) und als 3-Ebenen-Architektur realisiert. Damit wird eine Lastenverteilung auf drei Ebenen (Serverebene, Applikationsebene und Präsentationsebene) vorgenommen.
Obwohl diese Lastenverteilung grundsätzlich nur logischer Natur ist, ist es ratsam die Ebenen auch physikalisch zu trennen. Idealerweise besteht die Architektur aus einem dedizierten Datenbank-Server, der als Oracle-Portal Knoten (instanz von Oracle 8i in der 9iAS Portal installiert wurde) läuft. Dieser Knoten ist für die Kommunikation mit der Datenbank verantwortlich. In der Applikationsebene (middleware) befindet sich der Server mit 9iAS. In der Präsentationsebene befindet sich der Client, in diesem Fall ein Internet–Browser. Eine solche Aufteilung erleichtert die Wartung des Systems, bietet eine höhere Sicherheit und eine bessere Ressourcenverteilung.

Abb. 19: 3-Ebenen-Architektur (3–Tier)[7]

Wenn ein Benutzer eine Portal-Seite durch Eingabe der URL anfordert, ensteht folgender Fluß:

Die Abfrage geschieht aus dem Browser heraus und wird vom Oracle HTTP–Server abgefangen. Wegen dem „/pls“-Eintrag in der URL, weiß der Oracle HTTP–Server daß die Abfrage vom Mod_plsql-Modul weiterverarbeitet werden muß[8]. Mod_plsql führt den PL/SQL Code aus und benutzt die aktuelle DAD–Einstellung um die Abfrage an den Portal Knoten zu senden. Dieser beinhaltet die Seitendefinition, den HTML Code zur Anzeige der Seite und die Benutzereinstellungen.

Portlets werden über die Parallel Page Engine (teil vom 9iAS) geladen. Die Parallel Page Engine schickt die Abfragen an den Web Provider (Portlet Provider) weiter. Dies geschieht in mehreren parallelen Prozessen um die Geschwindigkeit zu erhöhen. Die Web Provider geben die Ausgabe der Portlets zurück an die Parallel Page Engine welche die Ergebnisse zu einer einzigen Seite zusammenführt und an den HTTP–Server weiterleitet. In der middletier–Ebene werden die Daten aus Performancegründen gecached.

[7] [Faeskorn, 2000]
[8] Vgl. [Pepper, 2001]

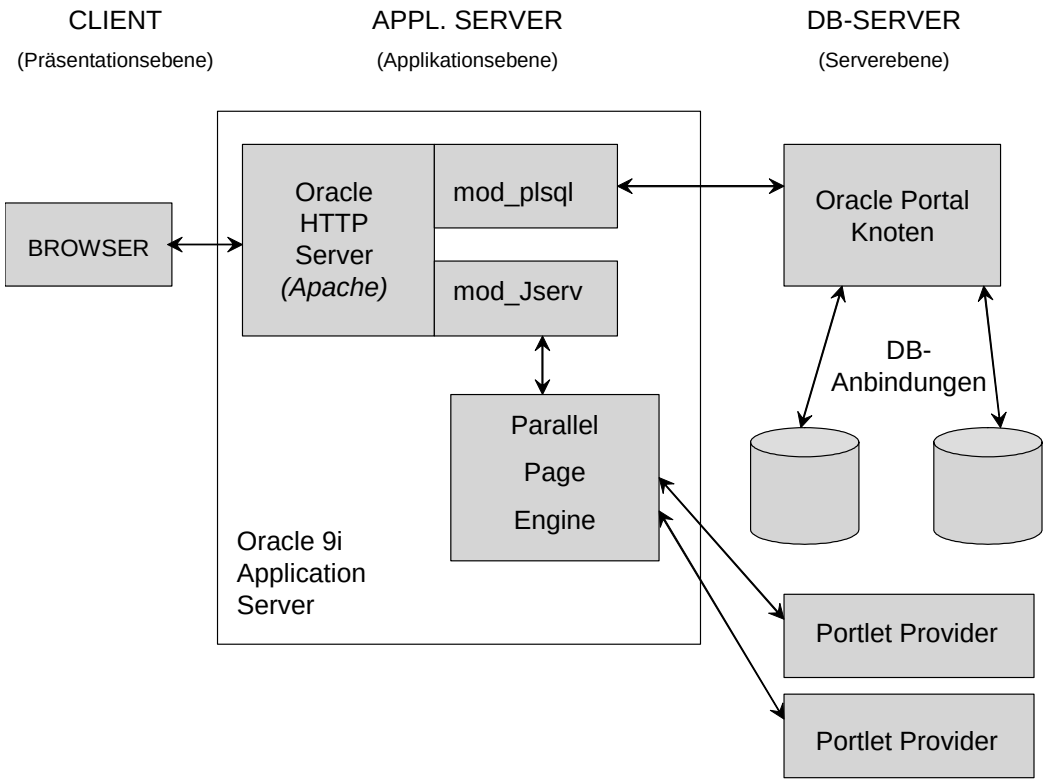

Abb. 20: Architektur des Oracle Portals [9]

Der HTTP-Server sendet letzlich die fertige HTML Seite an den Browser indem eine neue URL ermittelt und die Seite weitergeleitet wird. Welche Seite angezeigt wird ist benutzerdefiniert und wird über ein Cookie gesteuert, das beim Einloggen erzeugt wird.

3.2 Portlets und Pages

Die Portal-Elemente werden in sogenannte Portlets innerhalb von Pages organisiert.

3.2.1 Portlets

Ein Portlet ist ein Ausschnitt einer Portal–Seite, der als HTML-Tabellenzelle dargestellt wird und beliebige Daten beinhalten kann. Portlets sind wieder verwendbar und können aus einer Oracle Datenbank über Java oder PL/SQL oder aber aus einem Web–Provider über HTTP übermittelt werden.

[9] Vgl. [El-Mallah, 2002], S.10

3.2.2 Pages

Pages sind die eigentlichen Seiten des Portals. Sie beinhalten die Portlets und können selbst in Folders (Ordner) verwaltet werden.

3.3 Content Areas

Content Areas sind Container, die Oracle Portal Informationen (Pages und Portlets) beinhalten. Sie ermöglichen eine hierarchische Kategorisierung der Inhalte und die Vergabe von Rechten für die verschiedenen Benutzergruppen.
Die URL's der zu durchsuchenden Site müssen sich in einer solchen CONTENT AREA befinden, damit sie überhaupt indexiert werden können. (Eine Indexierung kann zwar aus SQL*Plus heraus auch für Elemente außerhalb der Content–Areas erzeugt werden, diese wird aber bei Suchanfragen aus dem Portal heraus nicht berücksichtigt). Das Einrichten einer neuen CONTENT AREA geschieht über die Schaltfläche CREATE NEW CONTENT AREA unter der BUILD Registerkarte in der Portal–Startseite (Abb.11).

Abb. 21: Einrichten einer neuen CONTENT AREA

Es können über einen Wizard bzw. Assistenten alle Einstellungen, die die Content Area betreffen (Sprache, Aussehen, Benutzerrechte), eingestellt werden.
Die ID der Content-Area wird später benötigt, sie kann entweder aus der Adressen-Leiste des Internet-Explorers abgelesen werden oder aus der Tabelle WWSBR_SITE$ ermittelt werden (hier 56).
Die Einrichtung einer Content Area kann auch über PL/SQL automatisiert werden, im Kapitel 6.2.1 wird die dafür nötige Funktion näher erläutert.

3.4 Kategorien

Es besteht die Möglichkeit die Inhalte innerhalb einer Content Area in verschiedene Kategorien aufzuteilen. So wäre es auch z.B. auch möglich, für die Subdomänen "Lehre", "Oracle", "Diplom", usw. jeweils eine eigene Kategorie einzurichten. Besonders sinnvoll lassen sich Kategorien in Verbindung mit XML–Dokumente erstellen. Dort können die Kategorien automatisch aus den XML–Tags erstellt werden. Ein Beispiel für die Funktionsweise der Kategorienerstellung mit XML – Dokumenten wird später unter Kapitel 7.1 erläutert.

4 Implementierung einer Spinne

Die Site, auf die die Suchmaschine wirken soll, ist zu groß um sie mit vertretbarem Aufwand manuell in eine Content-Area einzugeben. Es gibt aber bei den hier verwendeten Oracle-Produkten keine einfache Möglichkeit durch Eingabe einer URL alle mit der Seite verknüpften URL's zu speichern.
Es wird also ein externes Programm benötigt, welches bei Eingabe einer Startseite (seed–Adresse), diese Seite lädt, nach Verküpfungen untersucht und sich für jeden gefundenen Link selbst rekursiv aufruft (Spinne oder Crawler). Dieses Programm soll die URL's der sich in der Website befindlichen Textdokumente ausfiltern und einem ein für Oracle verwertbares Format ausgeben können.
Für künftige 9iAS-Versionen bietet sich dafür u.U. das Oracle9i-Werkzeug Ultra Search an, das für die hier verwendete Datenbank noch nicht zur Verfügung stand[10].

4.1 Programmimplementierung

Die Programmierung der Spinne lässt sich am einfachsten mit Java realisieren. Java bietet die Funktionen für die Behandlung von URL-Objekten von Haus aus an und ist multithread-fähig. Letzteres ist für das Crawlen durch die Seitenhierarchie sehr nützlich.
Das java.lang-Paket stellt eine Thread Klasse zur Verfügung, die Methoden für das Starten, Ausführen und Beenden eines Threads enthält und den Status des Threads überwacht[11].
Das Programm wurde als Applikation implementiert. Ein Applet ist wegen der Abschottung durch die Sandbox nicht so einfach zu realisieren, da es nur mit dem Server kommunizieren kann aus dem es aufgerufen wurde. Als Applikation besteht diese Einschränkung nicht. Die Ausgabe der Applikation auf die Standardkonsole lässt sich zudem sehr einfach in eine Datei umleiten. Weiter lässt sich durch die Verwendung von Java die Anwendung plattformunabhängig kompilieren und einsetzen. Von einem Laden der Klassen in die Datenbank wurde hier, obwohl prinzipiell durch entsprechende PL/SQL-Funktionen möglich, abgesehen.

[10] Vgl. [UltraSearch]
[11] Vgl. [Flanagan,1996] S.10

Als Basis für das Programm diente ein von Javaworld publizierter Artikel „Automating Web exploration“ von Laurence Vanhelsuwé[12], dessen Beispielcode dem Zweck entsprechend abgewandelt wurde.
Das Programm besteht aus vier Klassen und lässt sich im Pseudocode folgendermaßen beschreiben:

1. Lade die Seite (beim ersten Mal die mit der Seed–Adresse)
2. Extrahiere alle URLs aus der Seite
3. Für jede URL rufe das Programm rekursiv als Thread auf.

Die Klasse java.util.hashtable, welche die Datenstruktur einer Hashtabelle implementiert, wird benutzt um eine Datenbank mit bereits erfassten URL’s anzulegen, in der geprüft werden kann, ob eine URL bereits geladen wurde.
Das Programm wurde erweitert um URLs in Frames zu finden und um Verweise auf „nicht–HTML“-Dokumente aufzuspüren. Des weiteren wird durch das Programm automatisch eine Scripdatei für das Einlesen mit SQL*Plus erzeugt.

Die Syntax der erzeugten Scriptdatei wird in Kapitel 6 näher erläutert. Da das Programm hier eine zentrale Rolle spielt, wird der Quellcode der Klassen komplett komentiert in je einem Kapitel vorgestellt. Abbildung 22 zeigt ein Klassen-Diagramm des Programms.

[12] Vgl. [Vanhelsuwé, 1996]

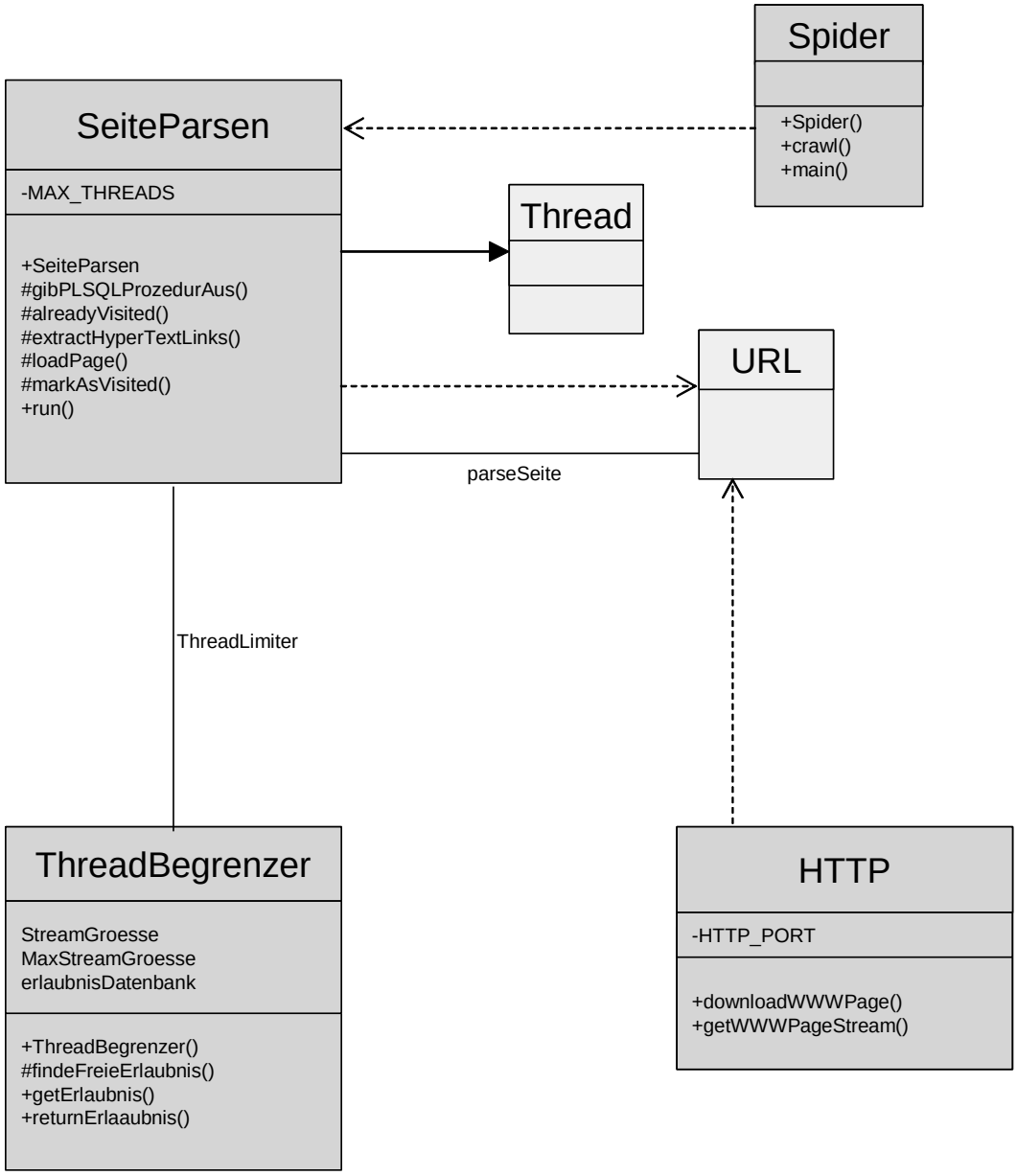

Abb. 22: Spider: UML - Diagramm

4.1.1 Die Klasse Spider

Die Klasse Spider nimmt die seed-Adresse `http://www.gm.fh-koeln.de/~faeskorn/` als Argument auf und instaziiert ein Thread–Objekt der Klasse SeiteParsen.

```
public class Spider
```

```
{  public Spider()  {  }
      public static void main(String[] args)
            {    new Spider().crawl(args);   }
      public void crawl (String[] args)
            {   new SeiteParsen( args[0] );   }}
```

4.1.2 Die Klasse SeiteParsen

SeiteParsen bilden den algorithmischen Kern des Programms. Diese Klasse untersucht die Struktur der von der Klasse `HTTP` übermittelten Zeichenkette (die komplette HTML-Seite), um nach weiteren Verküpfungen zu suchen und ruft für jede gefundene Verknüpfung die Klasse rekursiv auf. Die gefundenen Verweise werden in einer Tabelle gespeichert, um prüfen zu können, ob die Verkünpfung schon einmal vorgekommen ist. Dafür wurde eine statische Klasse (static) `Hashtable` definiert (statisch damit alle Threads Zugriff auf die gleiche Datenbank haben).
Die Klasse SeiteParsen wartet auf die Erlaubnis der statischen Klasse `Threadbegrenzer,` um einen neuen Thread zu starten.
Die Klasse URL ist Teil des java.net Pakets und erlaubt, dass die Daten, auf die URL verweist, heruntergeladen werden können. Aus der übermittelten Zeichenkette `Adresse` wird ein solches URL-Objekt instanziiert und der Thread (Klasse `run`) wird gestartet.

Die Methode `loadpage()` benutzt die Klasse `http,` um mit dem Webserver zu kommunizieren und die Webseite herunterzuladen.
Über die Methode `extractHyperTextLinks()` werden die auf der Seite enthaltenen URL's extrahiert und in einem Vector (Feld dynamischer Größe) gespeichert.

Ein Verweis (Hot Link) wird im HTML–Code zwischen den Anchor (Anker)–Tags <A> und </A> eingebettet. Beispiel: `<A HREF="http://...url...">..Seite...</A>`. Verweise auf Frames werden nicht mit dem sog. hypertext reference keyword `HREF` sondern mit `<FRAME SRC="http://...url...">` definiert[13].

[13] Vgl. [Münz, 2001]

Da das Parsen der Seite case sensitive ist und Frames enthält, wird sowohl nach den Zeichenketten „HREF=“ und „SRC=“ wie nach „href=“ und „src=“ gesucht.
Wenn die o.g. Zeichenketten als Teil des Textes statt als HTML-Tags auftreten, könnten theoretisch durch das Programm fehlerhafte URL’s abgefangen werden. Diese fehlerhaften URL’s werden später beim Hochladen als solche erkannt, so dass diese mögliche Fehlerquelle hier nicht abgefangen wird.

Relative Verweise (auf Seiten, die sich auf dem gleichen Server befinden) werden noch bearbeitet, indem der String `http://www.gm.fh-koeln.de/~faeskorn/` vorangestellt wird.

Die Konstante `MAX_THREADS` steuert die maximale Anzahl an gleichzeitig laufenden Threads.

Die Klasse Thread ist in Java bereits implementiert (java.lang.Thread). Um einen Thread zu erzeugen, muß eine Methode run() definiert werden. Die Methode run() beginnt mit der Ausführung, wenn die Start() Methode des Thread-Objektes aufgerufen wird. Mit sleep() wird der aktuelle Thread für eine angegebene Zeit angehalten.[14]

Für jede gefundene URL wird eine PL/SQL Prozedur auf die Standardkonsole ausgegeben, deren genaue Syntax später besprochen wird. Die Ausgabe der PL/SQL-Prozeduren wurde in eine eigene Methode `gibPLSQLProzedurAus()` implementiert

```
import java.util.*;
import java.io.*;
import java.net.*;
public class SeiteParsen extends Thread
{
private final static int MAX_THREADS = 5;
static ThreadBegrenzer threadLimiter = new ThreadBegrenzer(
MAX_THREADS );
static Hashtable seitenDatenbank = new Hashtable();
URL parseSeite;
```

[14] Vgl. [Flanagan], S.337

```
  public SeiteParsen(String Adresse)
  {
    try {
        parseSeite = new URL(Adresse);
        setName(Adresse);
        gibPLSQLProzedurAus(Adresse);
        start();     // Start des Threads mit run()
        } catch (MalformedURLException badURL) { ; }
  }
public void run() {
// Ein Thread wird nur gestartet, wenn er eine Erlaubnis bekommt
int erlaubnis;
// die komplette Webseite in eine Zeichenkettenvariable:
String webSeite;
// Vektoren, die die gefundenen URLs beinhalten:
  Vector seitenLinks;
  Vector seitenLinksSrc;
  Vector seitenLinksREF;
  Vector seitenLinksSRC;
//Die Adresse der Seite, die auf die aktuelle Seite verweist:
String mSeite;
//Zeichenketten, die auf eine URL deuten:
String lTyp;
    erlaubnis = threadLimiter.getErlaubnis();
    //Inhalt der Seite in String:
    webSeite     = loadPage(parseSeite);
    //URL:
    mSeite= "" + parseSeite;
// Der Seiteninhalt wird nach 'ref="' und 'src="' durchsucht:
    lTyp="ref=\"";
    seitenLinks   = extractHyperTextLinks(webSeite, mSeite, lTyp);
    lTyp="src=\"";
    seitenLinksSrc   = extractHyperTextLinks(webSeite, mSeite, lTyp);
// Nochmal mit Großbuchstaben:
    lTyp="REF=\"";
    seitenLinksREF   = extractHyperTextLinks(webSeite, mSeite, lTyp);
    lTyp="SRC=\"";
    seitenLinksSRC   = extractHyperTextLinks(webSeite, mSeite, lTyp);
    Enumeration enumREF = seitenLinksREF.elements();
    while(enumREF.hasMoreElements()) {
        String page = (String) enumREF.nextElement();
```

```
        if ( ! alreadyVisited(page) ) {
            markAsVisited(page);
                        new SeiteParsen( page);
        } else {;}
    }
    Enumeration enumSRC = seitenLinksSRC.elements();
    while(enumSRC.hasMoreElements()) {
        String page = (String) enumSRC.nextElement();
        if ( ! alreadyVisited(page) ) {
            markAsVisited(page);
                        new SeiteParsen( page);
        } else {;}
    }
        Enumeration enum = seitenLinks.elements();
    while(enum.hasMoreElements()) {
        String page = (String) enum.nextElement();
        if ( ! alreadyVisited(page) ) {
            markAsVisited(page);
                        new SeiteParsen( page);
        } else {;}
    }
  Enumeration enumSrc = seitenLinksSrc.elements();
    while(enumSrc.hasMoreElements()) {
        String page = (String) enumSrc.nextElement();
        if ( ! alreadyVisited(page) ) {
            markAsVisited(page);
                new SeiteParsen( page);
        } else {;}
    }
    // Die Erlaubnis wird nun zurückgegeben:
        threadLimiter.returnErlaubnis(erlaubnis);
    // Hier wird eine gewisse Zeit gewartet um die Thread-Ausführung
    // und somit die Erlaubnisabgabe willkürlirch anzuordnen:
     try {
        Thread.sleep( (int) (Math.random()*200) );
    } catch (Exception e) {}
}
//-----------------------------------------------------------------
// Aus einer URL, gibt die Seite als Zeichenkette zurück.
//-----------------------------------------------------------------
protected String loadPage(URL page) {
```

```
HTTP http;
    http = new HTTP();
    return http.downloadWWWPage(page);
}
//-----------------------------------------------------------------
// Aus der Zeichenkette, die eine HTML-Seite beinhaltet, extrahiere
// alle URL's und gebe die Liste als Zeichenketten-Vektor zurück.
//-----------------------------------------------------------------
protected    synchronized    Vector    extractHyperTextLinks(String
page,String mSeite, String lTyp) {
int letztePosition = 0;// Position vom "ref=" substring in der Seite
int endOfURL;                   // Endposition von http://........
String link;                    // Link dass verarbeitet wird
Vector  linkMenge  =  new  Vector();   // Vektor  um  gefunden  Links  zu
speichern
String dirExt="";
int positionLetzterSlash;
int positionErsterSlash;

    while(letztePosition != -1 ) {
        letztePosition = page.indexOf(lTyp, letztePosition);
        if (letztePosition != -1) {

            endOfURL = page.indexOf("\"", letztePosition + 5 );

 // extract found hypertext link
            link =  page.substring(letztePosition + 5, endOfURL);
            link = link.trim();
positionErsterSlash = mSeite.indexOf("/", 0);
positionLetzterSlash = mSeite.lastIndexOf("/");
if (link.indexOf("http://www.gm.fh-koeln.de/~faeskorn/",0)!=-1 )
    { dirExt= "";
//Link fängt mit HTTP://www... an }
if (link.indexOf("http://www.gm.fh-koeln.de/~faeskorn/",0)==-1 )
    { // Link fängt nicht mit HTTP://www... an
      dirExt    =    "http:"+    mSeite.substring(positionErsterSlash,
positionLetzterSlash)+"/"; }
if (link.indexOf("../")!=-1 )
    {  //Link fängt mit ../ an
      link=link.substring((link.indexOf("/",0)+1),(link.length()));
```

```
      dirExt=      "http:"+      mSeite.substring(positionErsterSlash,
positionLetzterSlash);
      dirExt= dirExt.substring(0,dirExt.lastIndexOf("/"))+"/";
        }
        link = dirExt  + link;
     //Zeichenkette wird formatiert:
        String linkerTeil;
        String rechterTeil;
        while (link.indexOf("/..")!=-1)
        {
        linkerTeil=link.substring(0,link.indexOf("/..")-1);

linkerTeil=linkerTeil.substring(0,linkerTeil.lastIndexOf("/"));

rechterTeil=link.substring(link.indexOf("/..")+3,link.length());
        link=linkerTeil+rechterTeil;
        }
        link = link.trim();
                if (link.endsWith("\"")) {
                link = link.substring(0, link.length() - 1 );
            }
            // Referenzen innerhalb derselben Seite werden ignoriert:
            if (link.indexOf("#") != -1) {
                link = link.substring(0, link.indexOf("#"));
            }
           if (link.endsWith(".gif") || link.endsWith(".jpg"))
           {
           markAsVisited(link);
           }
          // Hier kann entschieden werden welche Art von Dokumenten
(außer html)
          //  in die Liste aufgenommen werden:
       if (link.endsWith(".pdf") || link.endsWith(".doc") ||
            link.endsWith(".xls") || link.endsWith(".mdb") ||
            link.endsWith(".zip") || link.endsWith(".ppt") ||
            link.endsWith(".PDF") || link.endsWith(".DOC") ||
            link.endsWith(".XLS") || link.endsWith(".MDB") ||
            link.endsWith(".ZIP") || link.endsWith(".PPT"))
            { if ( ! alreadyVisited(link) )
                {
                  markAsVisited(link);
```

```
                gibPLSQLProzedurAus(link);
               }}
            // Folgende Links werden ignoriert:
            else
            if (link.indexOf("/http:")!=-1)
            {  //Es liegt außerhalb des Domains }
            else
            if (link.indexOf("mailto:")!=-1)
            {  //Link auf eine email - Adresse }
            else
             if (link.indexOf("javascript")!=-1)
            { //Javascript - Link }
            else
             if (link.indexOf("?")!=-1)
            {//Dieser Link übergibt Parameter an einen Script}
            else
             if (link.indexOf("&")!=-1)
            {  // Dieser Link übergibt Parameter an einen Script}
                else
            {   linkMenge.addElement( link );  }
        letztePosition ++; //die aktuelle Position wird übersprungen
        }
    }
    return linkMenge;
}
//-------------------------------------------------------------
// Prüfe ob eine Seite bereits gefunden wurde
//-------------------------------------------------------------
protected boolean alreadyVisited(String pageAddr) {
    return seitenDatenbank.containsKey(pageAddr); }
//-------------------------------------------------------------
// Markiere die Seite als bereits gefunden, indem sie in eine
// Datenbank eingefügt wird.
//-------------------------------------------------------------
protected void markAsVisited(String pageAddr) {
    seitenDatenbank.put(pageAddr, pageAddr); // fügt die Seite in die
                                             // Datenbank ein.
}
//-------------------------------------------------------------
// Gebe die PL/SQL – Prozedur aus
//-------------------------------------------------------------
```

```
protected void gibPLSQLProzedurAus(String Adresse)
{            System.out.println ("set serveroutput on size 1000000");
                  System.out.println ("declare");
                  System.out.println ("l_item_nr number;");
                  System.out.println ("l_site_id number;");
                  System.out.println ("l_corner_id number;");
                  System.out.println ("l_type_id number;");
                  System.out.println ("l_type_caid number;");
                  System.out.println ("l_region_id number;");
                  System.out.println ("l_hide_in_browse number;");
                  System.out.println ("begin");
                  System.out.println ("l_site_id := 56;");
                  System.out.println ("l_corner_id := 1;");
                  System.out.println ("l_type_id := 3;");
                  System.out.println ("l_type_caid := 0;");
                  System.out.println ("l_region_id := 5;");
                  System.out.println ("l_hide_in_browse := 1;");
System.out.println("l_item_nr  :=  wwsbr_api.add_item  (  p_caid  =>
l_site_id");
        System.out.println(", p_folder_id => l_corner_id");
        System.out.println(", p_display_name => '" + Adresse + "'");
        System.out.println(", p_type_id => l_type_id");
        System.out.println(", p_type_caid => l_type_caid");
        System.out.println(", p_region_id => l_region_id");
        System.out.println(", p_hide_in_browse => l_hide_in_browse");
        System.out.println(", p_url => '" + Adresse + "'");
        System.out.println(");");
                   System.out.println("commit;");
                  System.out.println("end;");
                  System.out.println("/");
   }
}
```

4.1.3 Die Klasse Threadbegrenzer

Threads sind parallel laufende Teile eines Prozesses. Sie unterscheiden sich von parallel laufenden Prozessen, indem sie auf einen gemeinsamen Speicherbereich zugreifen. Da für jede gefundene Verknüpfung ein neuer Thread gestartet wird, muß die Anzahl der gleichzeitig laufenden Threads begrenzt werden um Überläufe zu

vermeiden. Bevor eine Seite geladen wird, wird über die Klasse `ThreadBegrenzer` geprüft, wie viele Threads schon laufen. Wenn die voreingestellte Menge nicht überschritten wird, wird der Thread gestartet, ansonsten wird gewartet bis ein Thread beendet wird und eine Stelle freigibt.

```
public class ThreadBegrenzer  {
int erlaubnisDatenbank[];           // Feld mit Erlaubnissen
int StreamGroesse;                 // Aktuelle Größe des Streams
int maxStreamGroesse;             // maximale Groesse des Streams
//-----------------------------------------------------------------
// Eine maximale Größe des Streams gegeben, wird dieser als leer
// initialisiert und die Erlaubnisse als frei markiert.
//-----------------------------------------------------------------
public ThreadBegrenzer(int maxStreamGroesse) {
    this.maxStreamGroesse = maxStreamGroesse;
    StreamGroesse = 0;      // Aktuelle Groesse des Streams
    erlaubnisDatenbank = new int[maxStreamGroesse];
    for(int i=0; i < maxStreamGroesse; i++) {
// Setze alle Erlaubnis IDs als frei:
        erlaubnisDatenbank[i] = -1;   } }
//-----------------------------------------------------------------
// Wenn eine Erlaubnis frei ist, gebe sie dem Client und markiere
// sie als vergeben. Andernfalls warte mit der Ausführung des
// Threads bis eine Erlaubnis durch die Methode returnerlaubnis()
// frei wird.
//-----------------------------------------------------------------
public synchronized int getErlaubnis() {
    // falls der Stream zu groß ist, warte:
    while (StreamGroesse == maxStreamGroesse) {
        try { wait();}
catch (InterruptedException leaveUsAlonePlease) {}    }
     int erlaubnis = findeFreieErlaubnis();
  // Markiere Erlaubnis:
     erlaubnisDatenbank[erlaubnis] = erlaubnis;
    StreamGroesse++;  // Anzahl der Erlaubnisse wird inkrementiert
    return erlaubnis;}
//-----------------------------------------------------------------
// Eine Erlaubnis wird freigegeben:
//-----------------------------------------------------------------
public synchronized void returnErlaubnis(int erlaubnis)
{   erlaubnisDatenbank[erlaubnis] = -1;    // Gibt Erlaubnis frei
    StreamGroesse--;
// Startet einen Thread welcher auf Erlaubnis wartet:
    notifyAll(); }
//-----------------------------------------------------------------
```

```
// Findet freie Erlaubnis und gibt sie zurück:
//-------------------------------------------------------------
protected int findeFreieErlaubnis() {
    for(int i=0; i < maxStreamGroesse; i++) {
        if (erlaubnisDatenbank[i] == -1) {
            return i;        }    }
return -1; }}
```

4.1.4 Die Klasse HTTP

Die Klasse `HTTP` beinhaltet den Code für das Laden der kompletten HTML–Seite. Diese wird über das HTTP-Protokoll geladen, in eine Zeichenketten-Variable gespeichert und an die Klasse SeiteParsen übergeben.

```
import java.net.*;
import java.io.*;
class HTTP {
// Port 80 gilt als "well-known"-Port für HTTP:
public final static int HTTP_PORT = 80;
DataInputStream in;   // Den input stream für den HTML-Text
//-------------------------------------------------------------
// Durch Übergabe einer URL wird die entsprechende Seite geladen
// und in einer einzigen Zeichenkette gespeichert.
//-------------------------------------------------------------
public String downloadWWWPage(URL pageURL) {
String host, file;
InputStream pageStream = null;
    host = pageURL.getHost();
    file = pageURL.getFile();
    file= file.substring(1, file.length());
    try {
        pageStream = getWWWPageStream(host, file);
        if (pageStream == null) {return ""; }
    } catch (Exception error) {return ""; }
    DataInputStream in = new DataInputStream(pageStream);
    StringBuffer pageBuffer = new StringBuffer();
    String line;
    try {
 while ((line = in.readLine()) != null) {pageBuffer.append(line);}
    } catch (Exception error) { }
    try {        in.close();    }
      catch (Exception ignored) {}
    return pageBuffer.toString();}
//-------------------------------------------------------------
```

```
// Mit Übergabe des Hosts- und des Dateinamens, wird eine
// Verbindung ins Internet hergestellt,die HTTP-Antwort
// verarbeitet und das entsprechende Dokument geladen.
//-------------------------------------------------------------
public InputStream getWWWPageStream (String host, String file)
throws IOException, UnknownHostException {
Socket          httpPipe;   // Der TCP socket zum Web Server
InputStream     inn;       // Der raw byte input stream vom Server
OutputStream    outt;     // Der raw byte output stream zum Server
PrintStream     out;        // Den output stream als Text
InetAddress     webServer;  // Die Adresse des Web - Servers
    webServer = InetAddress.getByName(host);
    httpPipe = new Socket(webServer, HTTP_PORT);
    if (httpPipe == null) {  return null;  }
    inn  = httpPipe.getInputStream();    // Stream wird eingelesen
    outt = httpPipe.getOutputStream();
    in   = new DataInputStream(inn);    // Stream wird konvertiert
    out  = new PrintStream(outt);
    if (inn==null || outt==null) { return null; }
    // GET-HTTP-Anfrage wird gesendet:
    out.println("GET /" + file + " HTTP/1.0\n");
    // Antwort wird gelesen:
    String response;
    while ( (response = in.readLine()).length() > 0 ) {    }
    return in;      // InputStream wird zurückgegeben }}
```

4.2 JDeveloper

Als Entwicklungsumgebung für das Programm wurde Oracle9*i* JDeveloper Release Candidate 2 benutzt, es kann für Testzwecke kostenlos von den Oracle-Seiten heruntergeladen werden. Die Installation verläuft reibungslos. Die Arbeitsumgebung ist, wie bei modernen IDE's üblich, standardmäßig in drei Fenster aufgeteilt, eines für die Baumansicht des Projektes, eines für die eigentliche Codierung und eines für Status- und Fehlermeldungen. Nach Eingabe des Java-Codes in die jeweiligen Dateien können diese mit RUN->BUILD PROJECT zusammen kompiliert werden.
Um das Programm zu kompilieren und auszuführen reicht auch eine Kopie des Java SDK’s, welches kostenlos von http://java.sun.com geladen werden kann.

4.3 Programmaufruf

Das Programm kann nach der Kompilierung mit dem Befehl

```
java Spider http://www.gm.fh-koeln.de/~faeskorn/index_navigation_
menue.htm >loader.sql
```

gestartet werden. Je nach Verbindung kann das Programm mehrere Stunden benötigen, um die komplette Site zu durchcrawlen. Die fertige Datei LOADER.SQL ist über 5 MByte groß.

Für die Exklusion von Seiten durch Suchmaschinen hat sich als de facto-Standard mittlerweile eine Datei mit den Namen ROBOTS.TXT etabliert. Diese Datei wird von vom Hersteller der Seite im Web-Verzeichnis angelegt und auch von den meisten Suchmaschinen berücksichtigt. Sie enthält Informationen für das Harvesting über erlaubte und nicht erlaubte Seiten und Suchmaschinen[15].
Da das hier besprochene Programm die Existenz von Robotern ignoriert, sollte es auch nur mit eigenem Server oder mit der Erlaubnis des Betreibers verwendet werden.

[15] Vgl. [Koster, 1994]

5 Intermedia Text

InterMedia ist eine Gruppe von Datenbankerweiterungen, die den Umgang mit Bildern, Audio- und Videodateien sowie Dokumenten erleichtert. InterMedia Text ist der Teil von InterMedia der eine Suche in eine großen Menge von Dokumenten erlaubt[16]. In früheren Oracle-Versionen hieß diese Erweiterung ConText.

5.1 Die Oracle Text Architektur

Die zu indexierenden Dokumente durchlaufen sequentiell mehrere Ebenen bis zur eigentlichen Indexerstellung.

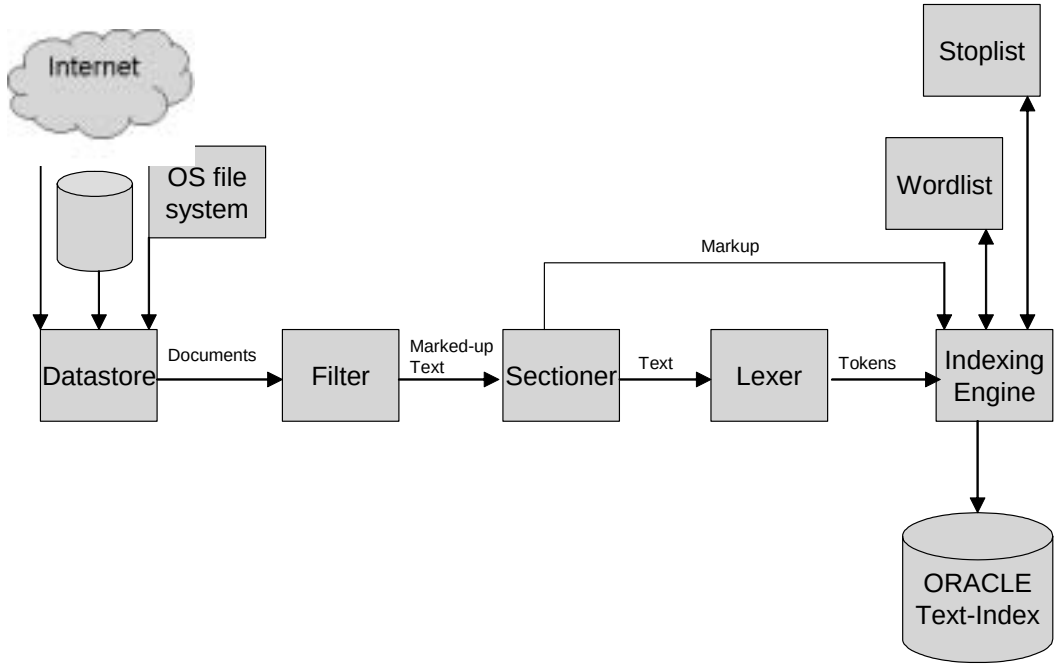

Abb. 23: Oracle Text Architektur [17]

[16] Vgl. [Muench, 2000], Kap. 13
[17] [Alonso, 2001], S. 6

5.2 Datastore

Dokumente aus Datenbanken, aus den lokalen File-System oder aus dem Internet werden als Datastore–Objekte gespeichert.
Für das Internet bietet das URL-Datastore Objekt die Möglichkeit, nur die URL als Verweis zum eigentlichen Objekt in der Datenbank zu speichern.
Da hier nicht die komplette Website in der Datenbank gespiegelt werden soll, bietet sich hier diese Art der Speicherung an. Um den Index zu erzeugen, werden später die tatsächlichen Dokumente geladen.

Die URL_DATASTORE–Objekte sind für Dateien konzipiert, die im Intra- oder Internet über die HTTP oder FTP–Protokolle oder lokal über das Dateisystem als FILE erreicht werden können.

Jede URL wird in ein Textfeld mit folgender Syntax gespeichert:

```
[URL:]<access_scheme>://<host_name>[:<port_number>]/[<url_path>]
```

wobei in der access_scheme Zeichenkette das Protokoll eingetragen werden kann (ftp, http oder file), z.B. `http://www.gm.fh-koeln.de/~faeskorn/oracle/sqlnet4.htm`

Die URL darf nur druckbare ASCII Zeichen beinhalten, nicht konforme Zeichen müssen in die *%xx*–Notation umgewandelt werden.

Das FTP–Protokoll kann auch mit Angaben von Benutzer und Passwort in der Form `ftp://username:password@ftp.hostname` verwendet werden.

Die URL_DATASTORE–Objekte können mit Parameter versehen werden. Bei der Indexierung über das Portal werden allerdings standardmäßig Defaultwerte eingesetzt.

Die URL_DATASTORE-Objekte haben folgende Attribute:

`timeout`	Timeout in Sekunden. Zulässig ist ein Wert zwischen 15 und 3600. Defaultwert ist 30.
`maxthreads`	Die maximale Anzahl an Threads die gleichzeitig laufen sollen. (Zwischen 1und 1024). Default ist 8.
`urlsize`	Die Maximale Länge der URL-Zeichenkette in bytes**.** Zulässig ist ein Wert zwischen 32 und 65535. Default ist 256.

maxurls	Die maximale anzahl an Zeilen dass der interne URL buffer für HTML Dokumente verwenden darf. Zulässig ist ein Wert zwischen 32 und 65535. Default ist 256. Die maximale Größe des Speicherbuffers beträgt in etwa 5 Megabyte und setzt sich aus den Werten maxurls * urlsize zusammen.
maxdocsize	Die maximale Dokumentengröße in Bytes. Erwartet ein Wert zwischen 256 und 2.147.483.647 bytes (2 Gigabytes). Default ist 2.000.000.
http_proxy	Hier kann der Hostname eines http proxy servers eingegeben werden in der Form „hostname:portnummer“. Dies kann notwendig sein wenn z.B. aus ein Intranet über eine Firewall auf das Internet zugegriffen wird.
ftp_proxy	Hier kann der Hostname eines ftp proxy servers in der Form „hostname:portnummer“ eingegeben werden.
no_proxy	Hier können bis zu 16 Domänen (durch Kommas getrennt) angegeben werden die kein Proxy Server benutzen sollen, wenn zum Beispiel für diese Domänen ein direkter Zugriff über ein Intranet möglich ist.

Tab. 2: URL_DATASTORE Attribute

5.3 Filter

Da nicht nur reine Textdokumente sondern auch andere Formate indexiert werden müssen, werden die Dokumente gefiltert. Oracle Text besitzt Filter für 150 Dateiformate und es besteht dazu noch die Möglichkeit eigene Filter zu implementieren. Als Output erzeugen die Filter Marked-up Text, bzw. HTML–Text. HTML erlaubt den Filtern den Erhalt von in Tags eingebetteten Informationen wie z.B. Titel oder Formatierungsoptionen.

5.4 Sectioner

Der Sectioner erlaubt eine Kategorisierung der Inhalte über HTML oder XML–Tags. Kapitel 7.1 enthält ein Beispiel für die Funktionsweise und deren Anwendung (Autosectioner).

5.5 Lexer

Aufgabe des Lexers ist es, den Output des Sectioners in einzelne Wörter umzuwandeln und dabei Sonderzeichen und Stoppwörter zu entfernen.

Die Wörter die für die Indexierung nicht sinnvoll sind, werden in sog. Stoppwort-Listen verwaltet. Diese Listen sind abhängig von der verwendeten Sprache.

Oracle verwendet solche Listen, deren Inhalt unter http://otn.oracle.com/docs/products/oracle8i/doc_library/817_doc/inter.817/a77063/astopsup.htm#1234 eingesehen werden kann.

Eigene Listen können mit CTX_DLL.CREATE_STOPLIST[18] erstellt werden, diese müssen aber bei der Indexerstellung als Parameter auch mit angegeben werden.
Während der Indexierung wird eine Spalte berücksichtigt, die die Dokumentsprache beinhaltet, und nur die Stoppwörter der jeweiligen Sprache werden von der Indexierung ausgeschlossen[19]. Durch die Angabe der Sprache für jedes Dokument ist es möglich, in mehrsprachigen Datenbeständen mit Berücksichtigung der jeweiligen Sprache zu suchen. Diese Eigenschaft wird als Multi-Lexer bezeichnet.
Während der Abfrage wird dann der LEXER der Sprache benutzt, die in die Suchmaske eingegeben wurde.
Obwohl der Lexer die Stoppwörter aus dem Text entfert, wird die Position der entfernten Wörter gekennzeichnet. In der Suche werden dementsprechend nur die Ergebnisse angezeigt, die an der Stelle des Stoppwortes auch irgendein Stoppwort beinhalten.

Die Voreinstellungen für den Lexer werden wärend der Installation von der Scriptdatei `sbrimtlx.sql` eingerichtet.

5.6 Indexing Engine

Die Indexing-Engine erzeugt den invertierten Index, der in der Datenbank gespeichert wird. Ein invertierter Index wird erzeugt indem für jedes Wort eine Liste der Dokumente, in denen das Wort auftritt, gespeichert wird.
In Kapitel 7 wird die Indexierung näher besprochen.

[18] [OTN A77063-01]
[19] [Kaminaga]

6 PL/SQL Funktionen

6.1 SQL*Plus und PL/SQL

SQL*Plus ist die primäre Schnittstelle zum Oracle–Server[20]. DIE SQL*Plus Oberfläche ermöglicht die direkte Eingabe von SQL und PL/SQL Befehlen sowie das Ausführen von vorgefertigten Scriptdateien. PL/SQL ist eine ORACLE-herstellerspezifische prozedurale Programmiersprache und als Erweiterung zu SQL konzipiert. Einige Programmkonstrukte werden als sogenannte Packages als Datenbankobjekte gespeichert. Einige hier relevante Funktionen dieser Packages werden weiter unten näher erläutert. Durch die Verbindung von SQL*Plus mit PL/SQL erhält man die Funktionalität einer mächtigen Programmiersprache.

Diese Schnittstelle wird hier benutzt, um die Objekte in die Content–Area des Portals einzufügen.

6.2 Das PL/SQL wwsbr_api-API Package

Ein Verändern der Tabellen, in denen die Portal-Objekte über SQL–Befehle geladen werden ist zwar prinzipiell möglich, da die Tabellen aber viele von Portal intern benötigte Felder beinhalten, lässt sich dies nicht wirklich realisieren. Das wwsbr_api–Package beinhaltet Funktionen, um trotzdem die Content–Areas und Items von außerhalb des Portals zu manipulieren[21].

6.2.1 Funktion Add_content_area

Die Funktion add_content_area generiert eine Content Area und gibt eine ID zurück[22]. Falls die Content Area nicht mit Hilfe des Wizards generiert werden will (Kap.3.3), kann sie mit Hilfe dieser Funktion erzeugt werden.

[20] [OTN: SQL*Plus]
[21] [Technet: API packages]
[22] [Technet : add_content_area]

Die Syntax lautet:

```
function add_content_area
(
    p_name varchar2,
    p_display_name varchar2,
    p_versioning varchar2 default VERSIONING_NONE,
    p_default_language varchar2 default 'us',
    p_root_folder_type number default 1,
    p_logo_filename varchar2 default null
)
return number;
```

Folgende Parameter[23] werden beim Aufruf der Funktion übergeben:

p_name	Der interne Name für die CONTENT AREA. Darf nicht mehrmals innerhalb einer Installation auftreten und darf nicht mehr als 60 Zeichen lang sein.
p_display_name	Der angezeigte Name der CONTENT AREA
p_versioning	Die Versionsverwaltung der Objekte der Content Area kann mit den Konstanten AUDIT, SIMPLE oder NONE definiert werden. Die Konstanten, die hier angegeben werden können sind: **VERSIONING_AUDIT –** Es wird automatisch eine neue Version erzeugt, jedesmal wenn ein Objekt upgedatet wird. **VERSIONING_SIMPLE –** Der Benutzer entscheidet, ob eine neue Version erzeugt wird. **VERSIONING_NONE –** Die Versionsverwaltung wird nicht automatisch gestartet. Der Benutzer kann die Versionsverwaltung manuell einschalten.
p_default_language	Die Defaultsprache für die Content Area. Sie kann nicht zu einem späteren Zeitpunkt geändert werden.
p_root_folder_type	Die FOLDER_TYPE_ID, die als Vorlage für das Stammverzeichnis der neuen Content Area benutzt wird.
p_logo_filename	Der optionale Dateiname mit Pfadangabe eines Logos.

Tab. 3: Add_content_area Parameter

Beispielhaft wird die Funktion folgendermasen aufgerufen:

```
 New_Content_Area_ID := wwsbr_api.add_content_area
(
  p_name => 'SUCHMASCHINE',
  p_display_name => 'Suchmaschine FH Gummersbach',
  p_versioning => wwsbr_api.VERSIONING_AUDIT,
  p_default_language => 'de',
```

[23] [Technet : add_content_area]

```
  p_logo_filename => '\usr\local\tmp\fh-koeln.gif'
);
```

6.2.2 Funktion Add_item

Diese Funktion wird im Rahmen dieser Arbeit für das Einfügen der URL's verwendet (Kapitel 6.4). Die PL/SQL Funktion ADD_ITEM[24] ist wie auch die Funktion Add_content_area Teil des WWSBR_API–Packages.

Diese PL/SQL–Funktion kann als Benutzer PORTAL30 ausgeführt werden. Die Rechte für das Ausführen von wwsbr_api–Funktionen können aber auch über einen mitgelieferten Script (`sbrapi.sql`) an einen anderen Benutzer erteilt werden.

Die Funktion add_item hat folgende Struktur:

```
function add_item
(
  p_caid number,
  p_folder_id number,
  p_display_name varchar2,
  p_type_id number,
  p_type_caid number,
  p_region_id number,
  p_display_option in varchar2 default FULL_SCREEN,
  p_category_id number default GENERAL_CATEGORY,
  p_category_caid number default SHARED_OBJECTS,
  p_perspectives g_perspectiveidarray
  default g_perspectiveidemptyarray,
  p_perspectives_caid g_caid_array
  default g_empty_caid_array,
  p_author varchar2 default wwctx_api.get_user,
  p_image_filename varchar2 default null,
  p_image_alignment varchar2 default ALIGN_BOTTOM,
  p_description varchar2 default null,
  p_keywords varchar2 default null,
  p_file_filename varchar2 default null,
  p_text varchar2 default null,
  p_url varchar2 default null,
  p_plsql varchar2 default null,
```

[24] [Technet. add_item]

```
  p_plsql_execute_mode varchar2 default null,
  p_plsql_execute_user varchar2 default null,
  p_app_component varchar2 default null,
  p_app_param_screen number default NO,
  p_folderlink_id number default null,
  p_folderlink_caid number default null,
  p_publish_date varchar2 default null,
  p_expire_mode varchar2 default PERMANENT,
  p_expiration varchar2 default null,
  p_master_item_id number default null,
  p_hide_in_browse number default NO,
  p_checkable in number default NO,
  p_parent_item_id number default null,
  p_attribute_id wwsbr_type.array default wwsbr_type.empty,
  p_attribute_caid wwsbr_type.array
  default wwsbr_type.empty,
  p_attribute_data_type wwsbr_type.array
  default wwsbr_type.empty,
  p_attribute_value wwsbr_type.array default wwsbr_type.empty
  )
return number;
```

Die Parameter im einzelnem[25] :

p_caid	Die ID–Nummer der Content Area, in die das Objekt eingefügt werden soll.
p_folder_id	Die ID des Ordners in den das Objekt eingefügt werden soll.
p_display_name	Der angezeigte Name des Objektes.
p_type_id	Die Typ–ID, um zu spezifiezieren welche Art von Objekt erzeugt werden soll. Die hier benutzten Konstanten können unter http://technet.oracle.com/products/iportal/files/pdk /plsql/doc/sdk23con.htm eingesehen werden.
p_type_caid	Die ID der Content Area für das Objekt-Typ.
p_region_id	Die ID einer Region innerhalb eines Folders, in die das Objekt eingefügt werden soll. Erlaubte Werte können der Tabelle WWSBR_ALL_FOLDER_REGIONS.ID[26] entnommen werden.
p_display_option	Die Display Option steuert die Art in der die Objekte angezeigt werden. Default ist FULL_SCREEN. Weitere Konstanten können unter http://technet.oracle.com/products/iportal/files/pdk/

[25] [Technet. add_item]
[26] [Technet: Content Area Views], Abschnitt 19

	plsql/doc/sdk23con.htm eingesehen werden
p_category_id number	Die ID der Kategorie, in der das neue Objekt erzeugt werden soll. Die Werte können in der Tabelle WWSBR_ALL_CATEGORIES.ID[27] gefunden werden.
p_category_caid	Die Content Area für die Kategorie die dem Objekt zugewiesen werden soll (entweder 0 oder der selbe Wert wie von p_caid)
p_perspectives	Ein Feld von Ansichts- (Perspective)–ID's, die den Objekten zugewiesen werden sollen. Erlaubte Werte befinden sich in WWSBR_ALL_PERSPECTIVES.ID und müssen sich in der Content Area 0 oder in der gleichen Content Area wie das neue Objekt befinden.
p_perspectives_caid	Ein Feld mit Content Area–ID's für die Ansichten in p_perspectives. Werte können unter WWSBR_ALL_PERSPECTIVES.CAID[28] gefunden werden.
p_author	Der Autor des Objektes. Default ist der Erzeuger des Objektes der mit wwctx_api.get_user ausgelesen wird.
p_image_filename	Der Name mit Pfadangabe eines Bildes, der für dieses Objekt angezeigt werden soll.
p_image_alignment	Ausrichtung für das angezeigte Bild in bezug zu den anderen angezeigten Objekten. Folgende Werte sind erlaubt: ALIGN_LEFT, ALIGN_RIGHT, ALIGN_TOP, ALIGN_BOTTOM, ALIGN_ABSOLUTE_MIDDLE, ALIGN_ABSOLUTE_BOTTOM, ALIGN_TEXT_TOP, ALIGN_MIDDLE, ALIGN_BASELINE
p_description	Beschreibung des Objektes.
p_keywords	Stichwörter für das Objekt.
p_file_filename	Name und absolute Pfadangabe einer Datei, die für dieses Objekt geladen werden soll. Dieser Parameter kann nur mit Objekten des Typs ITEM_TYPE_FILE oder ITEM_TYPE_ZIP_FILE angewendet werden.
p_text	Text für ein Text -Objekt. Gilt nur für Objekte des Typs ITEM_TYPE_TEXT.
p_url	Eine URL für das Objekt. Gilt nur für Objekte des Typs ITEM_TYPE_URL.
p_plsql	PL/SQL-Code für ein PL/SQL - Objekt. Gilt nur für Objekte des Typs ITEM_TYPE_PLSQL. Der Objekttyp kann der Spalte base_item_type der Tabelle WWSBR_ITEM_TYPES[29] entnommen werden.
p_plsql_execute_mode	Ausführungsmodus (execution mode)[30] für PL/SQL–Verzeichnisse. Erlaubt sind die Werte PUBLIC_SCHEMA, DB_USER und CREATOR_SCHEMA.

[27] [Technet: Content Area Views], Abschnitt 8
[28] [Technet: Content Area Views], Abschnitt 9
[29] [Technet: Content Area Views], Abschnitt 12
[30] [Technet: API constants]

p_plsql_execute_user	Der Benutzer für DB_USER im PL/SQL Ausführungsmodus.
p_app_component	Die ID PROVIDER_ID.PORTLET_ID für eine Applikation des Typs ITEM_TYPE_COMP.
p_app_param_screen	Steuert die Anzeige der Parameter bei Objekten des typs ITEM_TYPE_COMP.
p_folderlink_id	Die Verzeichnis-ID (folder_id) für das Verzeichnis dass mit ein Objekt des Typs ITEM_TYPE_FOLDER_LINK verküpft ist.
p_folderlink_caid	Die Content Area–ID für das Verzeichnis auf das das Objekt des Typs ITEM_TYPE_FOLDER_LINK zeigt.
p_publish_date	Das Datum an dem das Objekt publiziert werden soll.
p_expire_mode	Der Auslauf-Modus (expiration mode)[31]. Dieser Wert wird an p_expiration übergeben und gibt Auskunft über die Art in der der Wert von p_expiration interpretiert wird. Kann die Werte PERMANENT, EXP_NUMBER oder EXP_DATE erhalten.
p_expiration	Je nachdem welcher Wert p_expire_mode erhält: **PERMANENT**, Null, das Objekt verfällt nicht. **EXP_NUMBER**, Gültigkeitsdauer in Tagen des Objektes. **EXP_DATE**, das Verfalldatum im NLS_DATE format.
p_master_item_id	Wird benutzt, wenn eine neue Version des Objektes eingefügt wird. Der Wert entspricht dem Wert master_item_id des Objektes in WWSBR_ALL_ITEMS.MASTERID[32].
p_hide_in_browse	Steuert die Editierbarkeit der Objekte bezüglich View oder Browse–Modi (Abb.24-25).
p_checkable	Spezifiziert, ob das Objekt von Benutzern mit entsprechenden Rechten modifiziert werden kann.
p_parent_item_id	Wenn dieser Parameter benutzt wird, wird das Objekt vom Objekt dessen ID als p_parent_item_id übergeben wird, abgeleitet.
p_attribute_id	Ein Feld von Attribute-ID's.
p_attribute_caid	Ein Feld von ID`s von Content Areas. Muß dem Wert von p_caid entsprechen oder 0 sein.
p_attribute_data_type	Ein Feld von Objekttyp-Werten wie 'url', 'text', etc[33].
p_attribute_value	Ein Feld mit Attribut-Werten.

Tab. 4: Add_item Parameter

Die Funktion gibt eine master item ID zurück. Eventuelle Fehlermeldungen können den Oracle Technet–Seiten34 entnommen werden.

[31] [Technet: API constants]
[32] [Technet: Content Area Views], Abschnitt 5
[33] [Technet: Content Area Views], Abschnitt 11
[34] [Technet. add_item]

6.2.3 Funktion Add_folder

Analog zur ADD_CONTENT–Funktion generiert die Funktion ADD_FOLDER ein Verzeichnis innerhalb einer Content Area[35].

6.3 Die submit_search-Methode im wwsbr_search_api-Package

Die SUBMIT_SEARCH Methode ist im wwsbr_search_api-Package enthalten.
Sie generiert HTML–Seiten mit Suchergebnissen und wird mit folgender Syntax aufgerufen:

```
procedure submit_search
(
  p_search_terms in varchar2 default null,
  p_search_operator in varchar2 default FIND_ANY,
  p_caid in number default 0,
  p_current_caid in number default null,
  p_language in varchar2 default null,
  p_folder_id in number default null,
  p_folder_caid in number default null,
  p_item_type_id in number default null,
  p_include_child_folders in number default NO,
  p_category_id in number default null,
  p_perspective_id in number default null,
  p_search_for_type in varchar2 default ALL_OBJECTS,
  p_attribute_id wwsbr_type.array default wwsbr_type.empty,
  p_attribute_name wwsbr_type.array default wwsbr_type.empty,
  p_attribute_caid wwsbr_type.array default wwsbr_type.empty,
  p_attribute_data_type wwsbr_type.array
  default wwsbr_type.empty,
  p_attribute_operator wwsbr_type.array
  default wwsbr_type.empty,
  p_attribute_value wwsbr_type.array
  default wwsbr_type.empty,
  p_style_id in number default null,
  p_style_caid in number default null
  );
```

[35] [Technet: add_folder]

Diese Methode wird von den im Portal generierten Suchmasken (Kap. 7.5) automatisch aufgerufen. Eine nähere Erläuterung der Parameter kann den Oracle Technet–Seiten entnommen werden[36].

6.4 Einfügen der URL-Items in die Content-Area mit der LOADER.SQL–Datei.

"Loader.sql" ist die von der Spinne (Kapitel 4) erzeugte ASCII-Datei. Innerhalb dieser Script-Datei wird eine Prozedur pro Item aufgerufen wobei die Deklaration der Variablen für jedes Item wiederholt wird. Damit werden folgende Probleme behoben die beim Ausführen einer einzigen großen Prozedur auftreten würden:
Die maximale Größe einer von SQL*plus bearbeitbare Prozedur läßt sich nicht direkt ermitteln da alle Einstellungen der Felder "Optionen" direkten Einfluß darauf haben. Wenn aber nur eine Prozedur erzeugt würde, in der nach der Deklaration die Objekte nacheinander erzeugt würden, hätte zum einen SQL*plus eine Prozedur dieser Größe mit Sicherheit nicht mehr bearbeiten können. Zum anderen ermöglicht dieser modulare Aufbau der Datei dass URLs mit z.B. CGI-Abfragen die das Zeichen "=" beinhalten nur zum Abbruch der einzelnen Prozedur führen und somit auch nur dieses Objekt, das für die Datenbank auch nicht relevant wäre, ignoriert wird.
Als PORTAL30 in SQL*Plus eingeloggt, kann die Scriptdatei LOADER.SQL mit START [Laufwerk:\Pfad\] loader gestartet werden.
Eventuelle Fehler beim Einfügen der Objekte können danach mit dem wwbsr_... -Befehl angezeigt werden.
Das Einfügen geschieht über die Funktion ADD_ITEM. Die Parameter wurden schon in Kapitel 6.2.2 näher beschrieben.
Die vom Programm erzeugte Datei „loader.sql“ hat beispielhaft für die URL `http://www.gm.fh-koeln.de/~faeskorn/allgemeines/allgemeines.htm`,
folgende Struktur:

```
set serveroutput on size 1000000
declare
l_item_nr number;
l_site_id number;
l_corner_id number;
l_type_id number;
```

[36] [Technet: submit_search]

```
l_type_caid number;
l_region_id number;
l_hide_in_browse number;
begin
l_site_id := 56;
l_corner_id := 1;
l_type_id := 3;
l_type_caid := 0;
l_region_id := 5;
l_hide_in_browse := 1;
l_item_nr := wwsbr_api.add_item ( p_caid => l_site_id
, p_folder_id => l_corner_id
, p_display_name => 'http://www.gm.fh-
koeln.de/~faeskorn/allgemeines/allgemeines.htm'
, p_type_id => l_type_id
, p_type_caid => l_type_caid
, p_region_id => l_region_id
, p_hide_in_browse => l_hide_in_browse
, p_url => 'http://www.gm.fh-
koeln.de/~faeskorn/allgemeines/allgemeines.htm'
);
commit;
end;
/
```

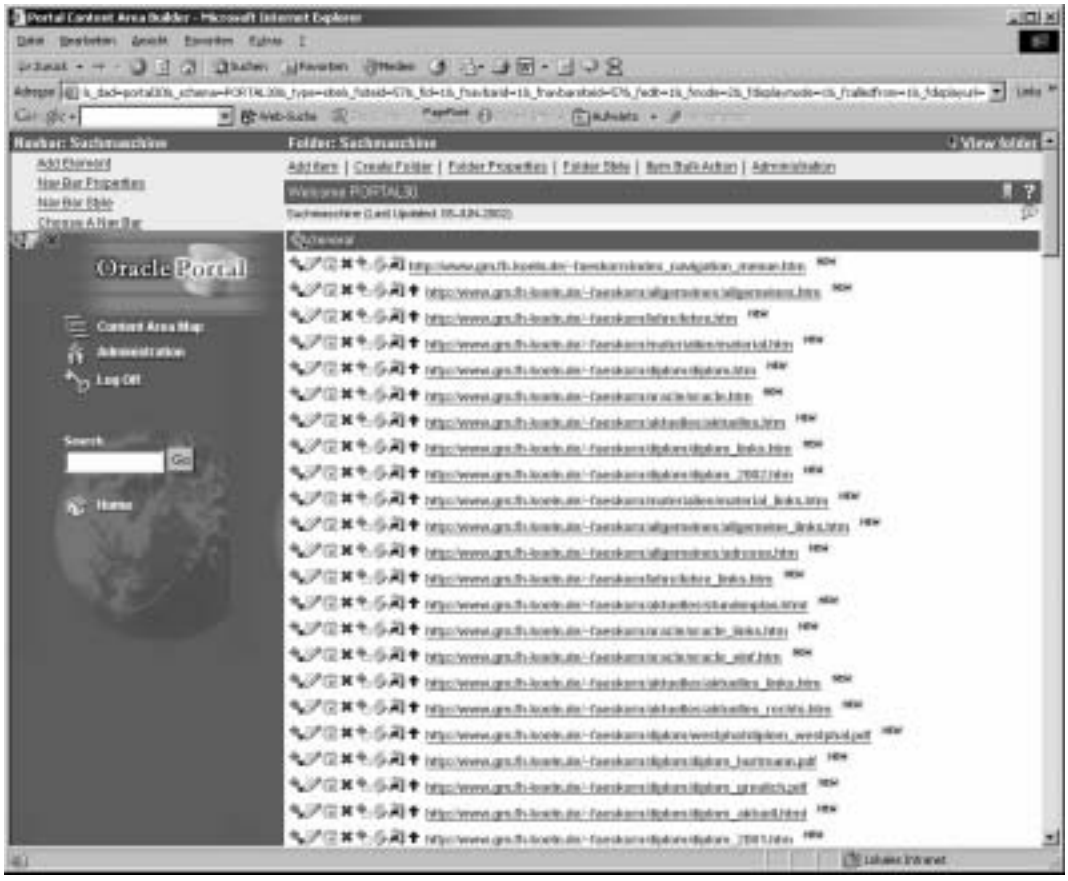

Abb. 24: Der Parameter p_hide_in_browse wurde auf 1 gesetzt

Der Parameter p_hide_in_browse wurde auf 1 gesetzt. Dies bewirkt dass die Objekte über den Browser editierbar bleiben (Abb.24). Ein Setzen des Parameters auf 0 blendet die Schaltflächen zum Editieren aus (Abb.25).

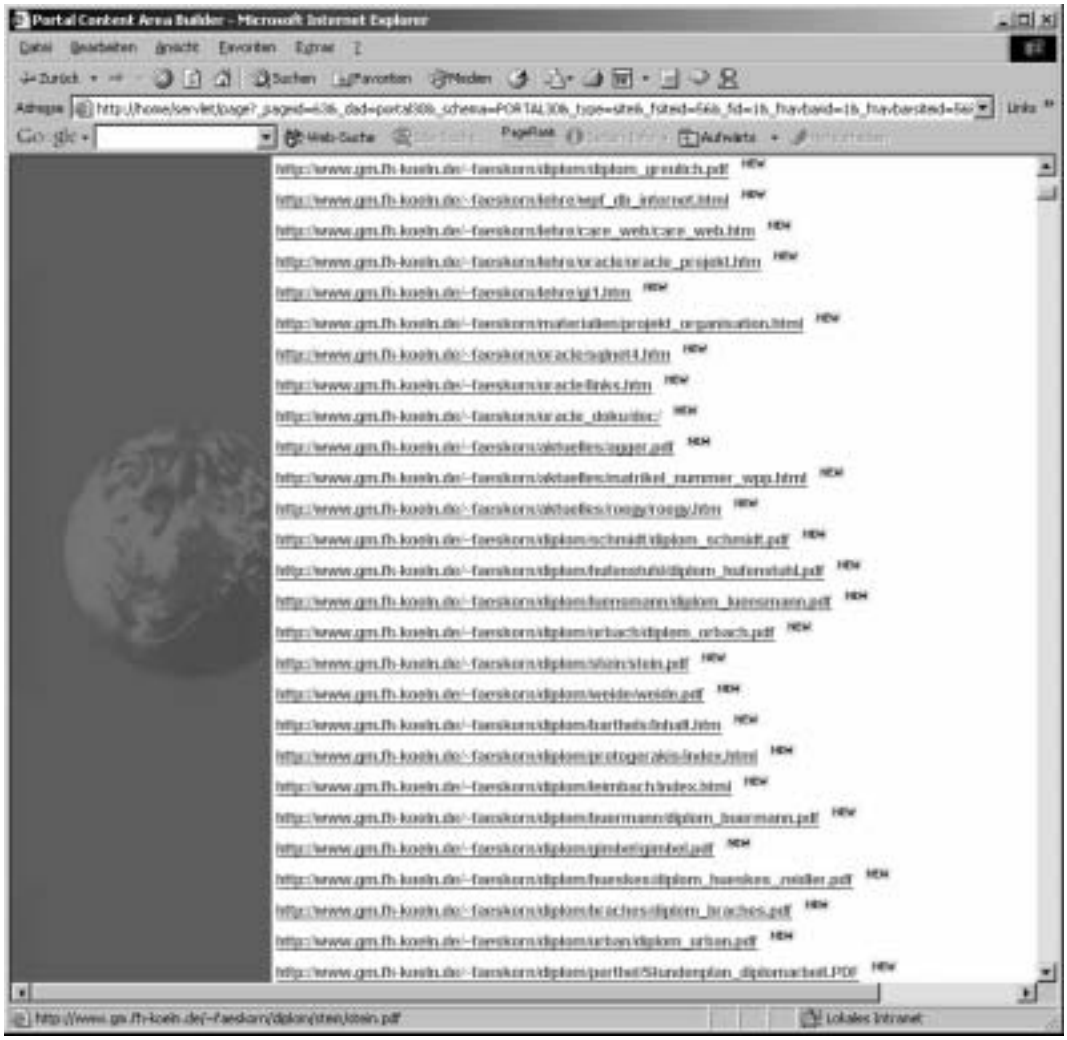

Abb. 25: Der Parameter p_hide_in_browse wurde auf 0 gesetzt

6.5 Laden der Objekte mittels iSQL*Plus

Alternativ zu SQL*Plus, lässt sich iSQL*Plus verwenden um die Daten auf ein nicht lokales Dateisystem einzuspielen. iSQL*Plus ist eine browserbasierte Schnittstelle zu SQL*Plus.

7 Suchfunktion mit InterMedia Text

7.1 Beispiel für das Suchen in XML–Dokumenten

Ein unter SQL*plus mit CREATE INDEX erzeugter Index wird im Portal als solcher erkannt, es erscheint auch die Schaltfläche DROP INDEX im Register Search. Leider wird dieser Index aber bei der Suchabfrage nicht berücksichtigt wenn die Tabelle nicht innerhalb einer Content-Area erzeugt wurde. Da die Tabelle, in der die URL's gespeichert werden viele Spalten mit Werten enthält, die vom Portal intern benutzt werden und nicht dokumentiert sind, ist es auch nicht sinvoll die Tabelle direkt zu manipulieren.
Als Beispiel für die Erzeugung und Funktionsweise eines CTX-Indexes soll aber trotzdem ein Beispiel ohne Verwendung des Portals dienen:
Es wird unter SQL*Plus eine Tabelle erzeugt mit:

```
CREATE TABLE xml_dokumente (
id NUMBER PRIMARY KEY,
dokumente CLOB
);
```

Die Spalte „dokumente“ beinhaltet XML-Dokumente (oder beliebig andere) mit bis zu 4 Gigabytes Text.
Ein Index wird erzeugt mit

```
CREATE INDEX dokumente_index ON xml_dokumente (dokumente) INDEXTYPE
IS ctxsys.context;
```

Der Context Index erlaubt nun ein Suchen innerhalb der Texte. Diese Abfrage gibt die ID’s der Dokumente aus, die den Suchbegriff enhtalten:

```
SELECT id
FROM xml_dokumente
WHERE CONTAINS (dokumente,  <Suchbegriff>)>0;
```

Im Zusammenhang mit XML kann im Index der AUTOSECTIONER benutzt werden (Falls schon ein Index existiert kann er mit `DROP INDEX dokumente_index;` gelöscht werden).
Ein Index wird erzeugt:

```
CREATE INDEX dokumente_index ON xml_dokumente (dokumente)
INDEXTYPE IS ctxsys.context
PARAMETERS ('section group ctxsys.auto_section_group');
```

Nun kann die Suche folgendermaßen erweitert werden:

```
SELECT id
FROM xml_dokumente
WHERE CONTAINS (dokumente ,<Suchbegriff> WITHIN <Kategorie>) > 0;
```

Nun werden die ID's der Dokumente die den Suchbegriff enthalten ausgegeben aber nur dann, wenn sie zwischen den Tags <Kategorie> und </Kategorie> eingebettet sind.

7.2 Der CTXSYS-Benutzer und die CTXAPP-Rolle

Alle Benutzer können einen interMedia Text Index erzeugen und benutzen. Für die Nutzung der PL-SQL Funktionen und Setzen der Einstellungen werden aber spezielle Rechte benötigt.

Der CTXSYS–User ist für die Administration von interMedia Text konzipiert und wird bei der Installation automatisch angelegt. Er kann System–Preferenzen ändern, Preferenzen von anderen Benutzer löschen und ändern, Prozeduren im CTX_ADM PL/SQL-package benutzen, den CTXSRV–Server starten, alle System-Views abfragen und hat alle Rechte eines Benutzers mit der CTXAPP–Rolle.

Die CTXAPP-Rolle ist für Anwendungsentwickler gedacht und erlaubt den Benutzern das Einrichten und Löschen von interMedia Text Preferenzen und das Benutzen der interMedia Text PL/SQL-Packages .

Zur Einrichtung und Nutzung der Funktionen rund um die Textsuche muss der Account also die CTXAPP–Rolle besitzen. Da die Textsuche in Oracle Portal eingebunden werden soll und dafür Oracle Portal Administrator–Rechte benötigt werden, wir der Account PORTAL30 oder ein Account welches die PORTAL30–Rolle besitzt benutzt. Es empfiehlt sich einen neuen speziellen Benutzer anzulegen da, wenn Portal30 aus Versehen Objekte löscht, es damit das ganze Portal zerstören könnte.
Eine bereits vorhandene Funktion kann verwendet werden um oben benannte Rechte automatisch zu vergeben. Sie wird unter SQL*Plus mit

```
execute wwv_context_util.grantCtxRole(USER)
```

aufgerufen[37].

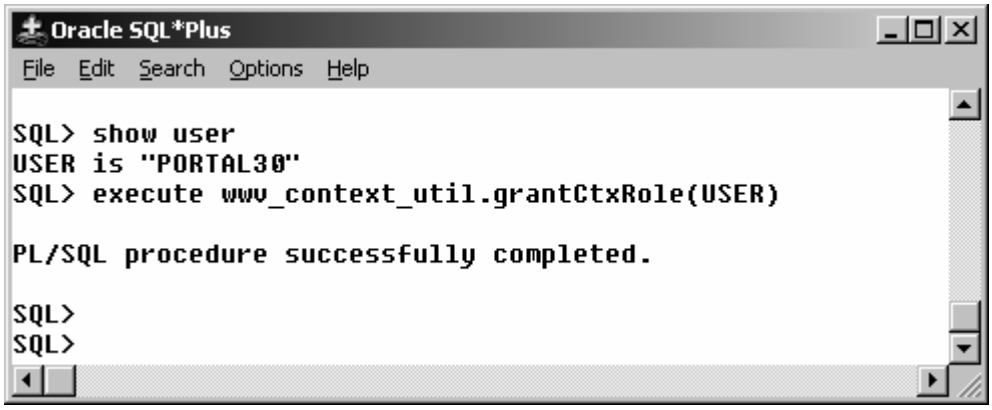

Abb. 26: Grant der ctx – Rechte für PORTAL30

7.3 Indexierung über das Oracle Portal

7.3.1 Gateway–Einstellungen

Für die Erzeugung eines Indizes unter Verwendung von Oracle Portal muss folgende Änderung im Gateway Konfigurationsmenü (http://<hostname>:port/pls/admin_/gateway.htm) vorgenommen werden:

[37] Vgl. [El-Mallah, 2002], S. 434

Das Connection Pooling bewirkt, dass eine bereits durch eine Anfrage geöffnete Verbindung nochmals für weitere Anfragen benutzt wird und nicht jedesmal geöffnet und heruntergefahren wird.

Unter Gateway Database Access Descriptor Settings muss aber die Einstellung “Enable Connection Pooling” auf NO gesetzt werden (Abb.27).

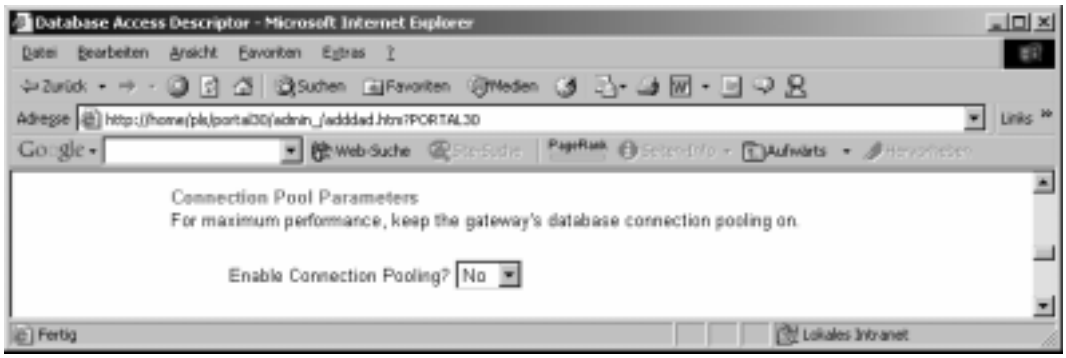

Abb. 27: Connection Pool Parameter

7.3.2 Indexerstellung

In der Registerkarte Administration der Portal Seite befindet sich ein Portlet “Search Settings” (Abb.28). Dort muss die Option ‚**Enable** *interMedia* Text **Searching**’ aktiviert werden.

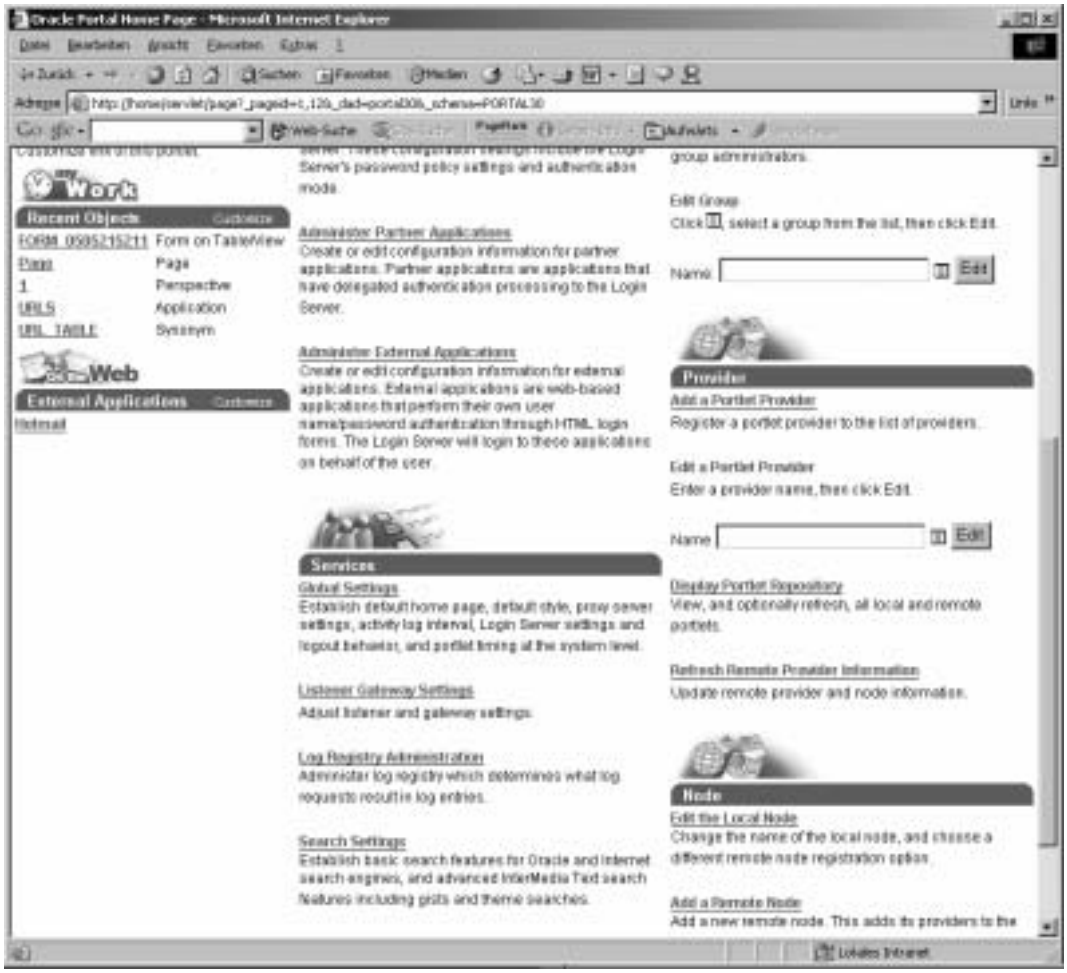

Abb. 28: Oracle Portal Home Page

Über die Listenfelder **Highlight Text Color** und **Highlight Text Style** kann das Aussehen der Suchergebnisse verändert werden.

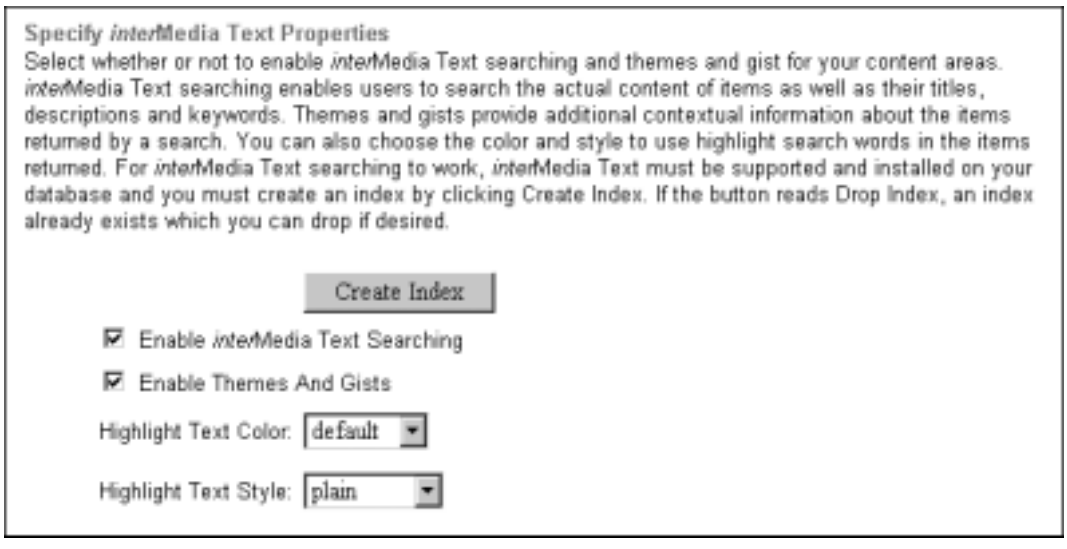

Abb. 29: Indexeigenschaften

Wenn **Enable Themes And Gists** aktiviert wird, werden bei der Suchausgabe noch folgende Informationen angezeigt:

Ein „Theme“ zeigt die Namen und Verben, die am öftesten in den Objekten auftreten und ein „Gist“ zeigt eine kurze Zusammenfassung der Objekte basierend auf der Häufigkeit, in der diese Namen und Verben auftreten.

Der Index kann mit einem Klick auf die Schaltfläche „CREATE INDEX“ erzeugt werden.
Während der Indexierung werden intern alle Dokumente konvertiert, so dass eine spätere Suche keine Performanceverluste durch Konvertierungszeiten mit sich bringt. Der Indexierungsprozeß benötigt je nach System mehrere Stunden. Die komplette Site ist ca. 540 MB groß und die Indexierungsgeschwindigkeit liegt zwischen 50 MB und 1 GB pro Stunde, wobei die Formatkonvertierungszeiten vernachlässigt werden können.
Wenn die Indexierung beendet ist, meldet SQL*Plus „`Index created`“.
Wenn bereits ein InterMedia-Index existiert, erscheint eine Schaltfläche „DROP INDEX“, mit der die alten Indexdateien gelöscht werden können.
Es werden für alle Objekttypen automatisch Indizes erzeugt. Es handelt sich dabei um invertierte Listen.
Für den URL_INDEX werden alle Seiten die durch ein URL–Objekt referenziert werden geladen und indexiert. Analog entstehen der TEXT_INDEX für Text–Objekte und FILE_INDEX für Datei–Objekte.
Des weiteren werden folgende Indexe erzeugt:
CONTEXT_AUTHOR_INDEX welcher auf den Autor der Objekte basiert
CONTEXT_DESC_INDEX , basierend auf der Beschreibung der Objekte.
CONTEXT_KEYWORDS_INDEX , basierend auf den benutzten Keywords.
CONTEXT_TITLE_INDEX, basierend auf dem Title der Objekte.
Der Index kann auch manuell unter SQL*Plus erzeugt werden, dazu muß folgende Scriptdatei ausgeführt werden:

`ctxcrind.sql` (Unter `<ORACLE_HOME>/portal30/src/wws`)

Sollten bei der Indexerstellung Probleme auftauchen oder falls interMedia Text neu installiert werden musste, kann folgende Scriptdatei unter SQL*Plus ausgeführt werden um die Einstellungen zu resetten:

`inctxgrn.sql` (Ebenfalls unter <ORACLE_HOME>/portal30/src/wws)

Der Erstellte Index ist ein CONTEXT – Index. Dies ist einer der 3 möglichen Oracle Intermedia Text–Indizes. Aus der folgenden Tabelle[38] wird ersichtlich warum ein CONTEXT–Index verwendet wird:

Index Typ	**Anwendung**	**Query – Operator**
CTXCAT	Für die Indizierung von kleine Textfragmente geeignet	CATSEARCH
CONTEXT	Für die Indexierung von langen koeherenten Dokumenten in verschiedenen Formaten wie z.B. Word, HTML, ASCII oder XML.	CONTAINS
CTXRULE	Ein Index für eine Tabelle mit Abfragen in der jede Abfrage klassifiziert wird.	MATCHES

Tab.5: Intermedia Indizes

7.4 Index–Aktualisierungen

Nachdem die Indizes erzeugt wurden, muss man sich Gedanken über eine etwaige spätere Aktualisierung machen. Je nachdem wie oft der Index aktualisiert werden soll, müssen verschiedene Strategien in Betracht gezogen werden. Da die Aktualisierung des Indexes ein ressourcenintensiver Prozeß ist muß die Wichtigkeit eines aktuellen Indizes gegenüber einer effizienteren Nutzung der Ressourcen gestellt werden.

[38]Vgl. [Alonso, 2001], S.11

7.4.1 Automatische Aktualiserung (Synchronisation)

Der Context Server wird mit ctxsrv gestartet und aktualisiert automatisch die Indexe, wenn ein Objekt zur Datenbank hinzugefügt wird. In Verbindung mit Oracle Portal sollte der Context Server aber nicht mehr verwendet werden.

Statt dessen sollte die ctx_schedule.startup–Funktion des ctx_schedule–Packages[39] benutzt werden. Um ctx_schedule zu installieren, muss die Scriptdatei ctx_schedule.sql[40] ausgeführt werden (eingeloggt als CTXSYS).

Folgende Kommandos starten den Sinkronisationsdienst:

```
exec ctx_schedule.startup ( 'url_index', 'SYNC', 10 ) ;
exec ctx_schedule.startup ( 'url_index', 'OPTIMIZE FAST', 60 ) ;
```

wobei die erste Zahl (10) der Anzahl an Minuten angibt, die zwischen den Aktualisierungen vergehen sollen. Die zweite Zahl (60) gibt die Minuten an, die zwischen den Optimierungen vergehen sollen. Im obigen Beispiel würde der Index URL_INDEX alle 10 Minuten aktualisiert und stündlich optimiert.

Die Indexaktualisierung kann mit den Kommandos:

```
exec ctx_schedule.stop ( 'url_index' ) ;
exec ctx_schedule.stop ( 'url_index', 'OPTIMIZE FAST' ) ;
```

beendet werden.

Wenn die Datenbank sehr vielen ständigen Änderungen unterworfen ist, kann das Script `drbgdml.sql`[41] (unter `$ORACLE_HOME/ctx/sample/script/`) verwendet werden. Diese Scriptdatei synchronisiert den Index alle 5 Sekunden.

[39] Vgl. [Metalink 132689.1]
[40] Vgl. [Metalink 132693.1]
[41] Vgl. [Portal A86707-01]

7.4.2 Manuelle Aktualisierung

Wenn, wie im Fall dieser Arbeit, die Änderungen der Website dem Administrator des Portals immer bekannt sind, kann u.U. auf eine ständige automatisierte Indexaktualisierung verzichtet werden.
Für die hier behandelte Suchmaschine reicht es aus, wenn der Index nach dem Einfügen von Objekten in die Content Area manuell aktualisiert wird, da neue Objekte auch manuell und nicht sehr oft eingefügt werden.

Dies geschieht durch Eingabe von

```
ALTER INDEX <indexname> REBUILD ONLINE PARAMETERS ('SYNC')
```

wobei die Indexnamen aus Kapitel 6 zu entnehmen sind.
Nachteil ist, dass die Fragmentierung der Indexe mit zunehmendem Gebrauch von „ALTER INDEX“ zunimmt. Deshalb empfiehlt Oracle den Befehl so wenig wie möglich einzusetzen[42].

Wenn sich eine sehr große Datenmenge geändert hat, kann es Sinnvoll sein den Index zu löschen und neu zu erstellen. Dazu muß im Services – Portlet unter Search Settings die Schaltfläche „DROP INDEX“ angeklickt werden (oder unter SQL*Plus `ctxdrind.sql` ausgeführt werden) und wieder wie in 6.2.4 beschrieben mit der Schaltfläche „CREATE INDEX“ ein neuer Index erzeugt werden.

7.5 Suchfunktion

Die Suchfunktion wird in drei Suchtypen unterteilt, wobei der letzte als Erweiterung der anderen beiden verstanden werden muß:

- Basic Search
- Advanced Search
- interMedia Search

[42] Vgl. [Portal A90096-01]

7.5.1 Basic Search

Diese Suchfunktion steht auch ohne interMedia Text zur Verfügung. Es handelt sich dabei um eine einfache Suchfunktion. Wenn interMedia Text nicht zur Verfügung steht, durchsucht diese Funktion keine Dokumente und keine Inhalte von URL's sondern, nur die Felder der Namen, Beschreibungen und Stichwörter (Keywords).

Abb. 30: Suchmaske für die BASIC-Search Funktion

Die Suchergebnisse werden in einen getrennten Portlet angezeigt, in dem auch die Möglichkeit einer erweiterten Suche (Advanced Search) angeboten wird.

7.5.2 Advanced Search

Advanced Search ist eine erweiterte Suchfunktion, die wie die Basic Search Funktion auch ohne interMedia Text im Portal zur Verfügung steht. Mit dieser Suchfunktion ist es möglich, die Suche auf mehrere Content Areas zu erweitern sowie auf bestimmte Objekttypen zu beschränken.

Für den Administrator des Portals besteht die Möglichkeit, die vorgegebene Advanced Search Seite durch eine eigene Seite ersetzen.

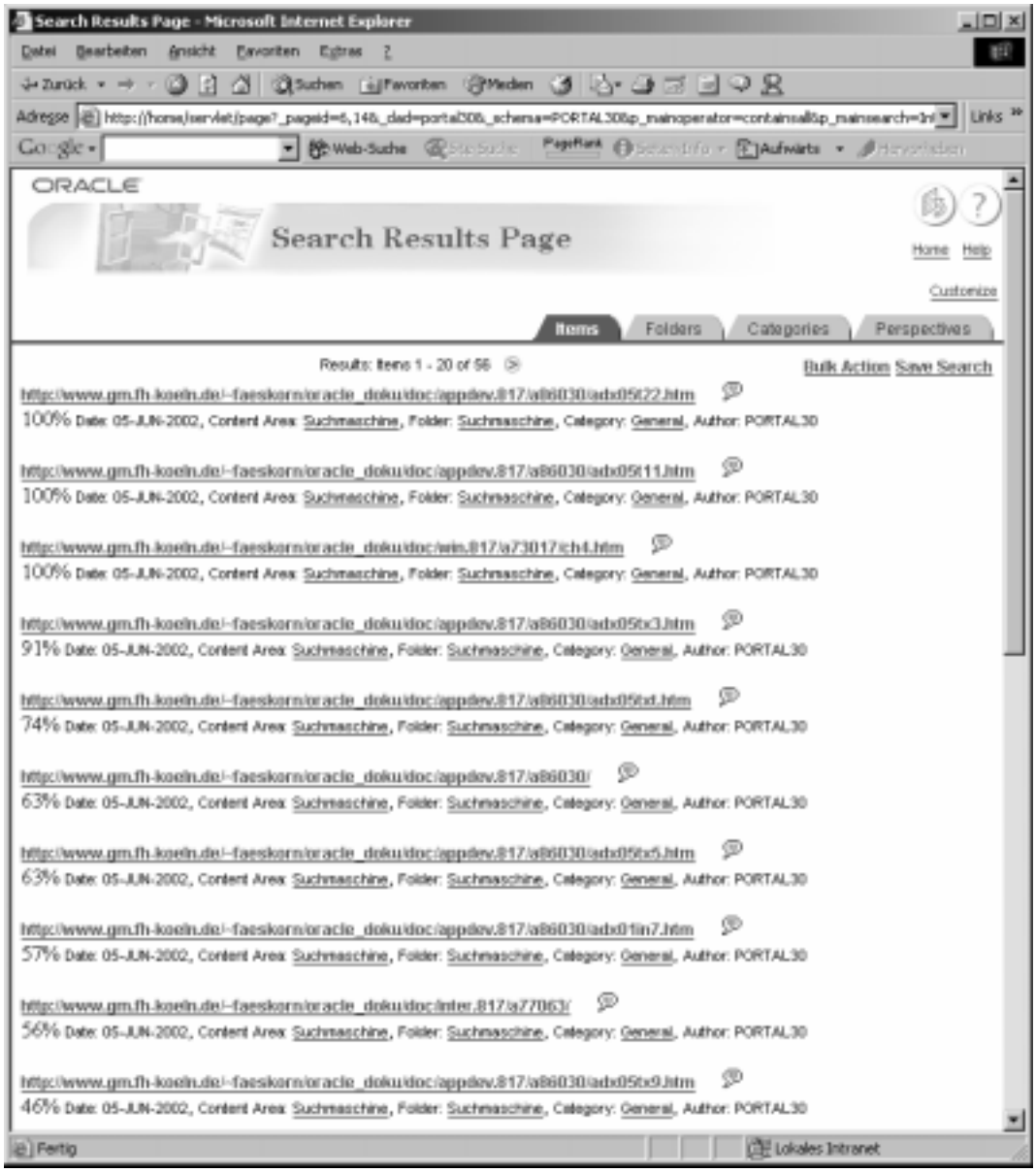

Abb. 31: Darstellung der Suchergebnisse

Die Suchergebnisse werden auch hier in einem getrennten Portlet aufgelistet. Dort ist es möglich, die gefundene URL's anzuklicken und in einem getrenntes Fenster zu öffnen.

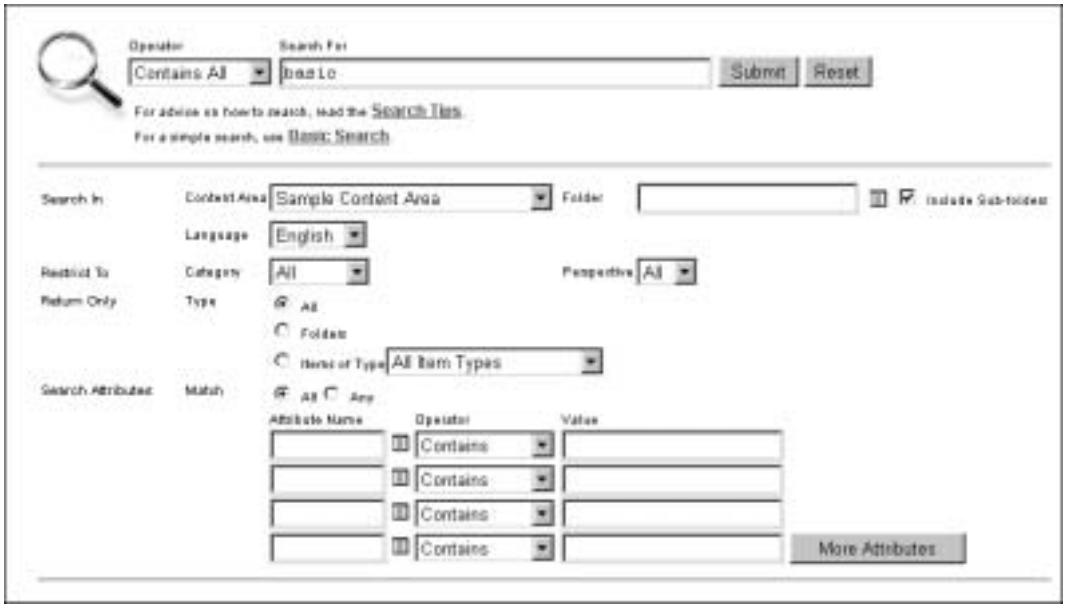

Abb. 32: Advanced Search Suchmaske

7.5.3 interMedia Search

Die interMedia-Text Suche im Oracle Portal funktioniert nur für die Content Areas. Die Suche erfolgt über das Oracle Portal Search Portlet.

Nach der Indexierung ist die Suche für alle Content Areas aktiviert. Sie kann nicht für einzelne Content Areas deaktiviert werden, obwohl die Sucherergebnisse nur für die jeweilige Content Area angezeigt werden, in der die Suche ausgeführt wurde.

InterMedia Search erweitert die Basic–und Advanced Search Funktionen und ermöglicht die Near, Stem, Soundex und Fuzzy–Suche.

Des weiteren wird die Suche ín Dokumenten ermöglicht (z.B. PDF, PowerPoint, Word, HTML, Text), auch wenn diese nur über ein URL–Objekt referenziert werden. Bei PDF–Dateien werden eingescannte Vorlagen nicht als Text erkannt sondern als Bitmaps behandelt.

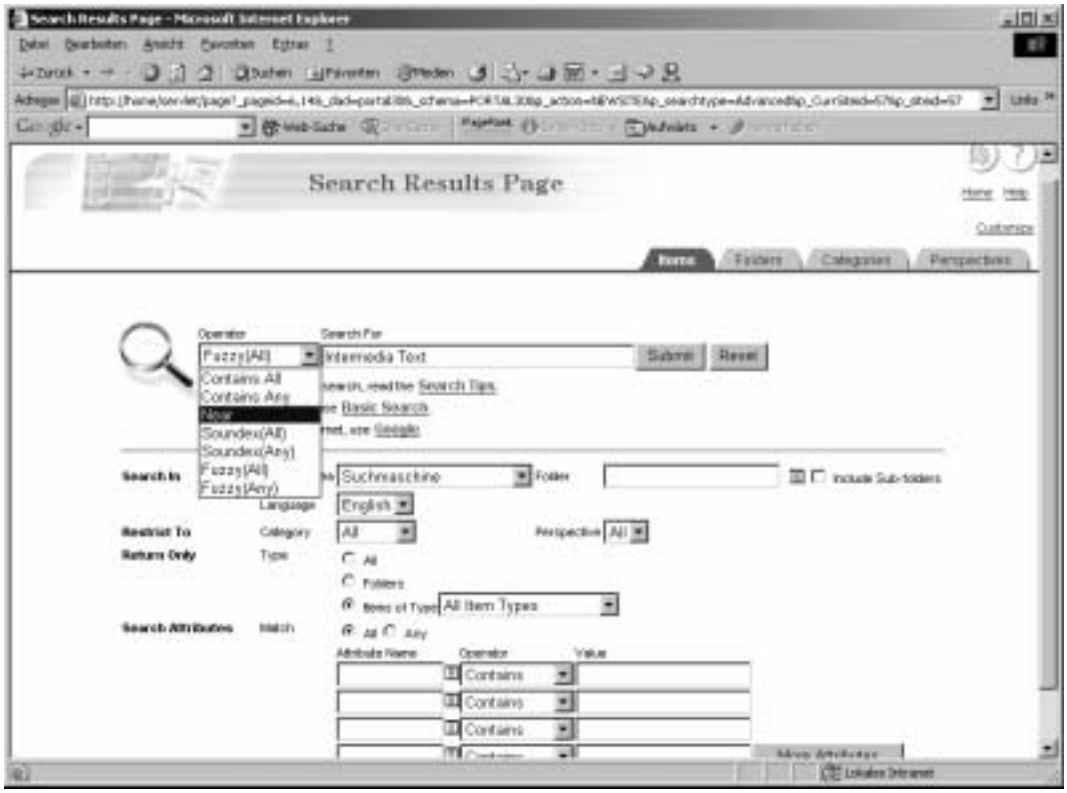

Abb. 33: Suchmaske für die interMedia Suche

7.6 Operatoren und erweiterte Suchfunktionen

Intermedia Text benutzt verschiedene intelligente Strategien um bestmögliche Ergebnisse zu erreichen. Man muß dabei beachten, dass einige Funktionen der Intermedia Textsuche nicht - oder nur Teilweise - über das Portal erreichbar sind.

7.6.1 Wildcards

Die Benutzung von Wildcards ist auch ohne interMedia möglich. Durch die Angabe des Zeichens „%“ als Platzhalter wird die Suche auf die Wörter erweitert, die ein beliebiges Zeichen an der Stelle des Platzhalters beinhalten. Dies ermöglicht das Suchen nach Begriffen, deren Schreibweise man nicht genau kennt.

7.6.2 CONTAINS ALL und CONTAINS ANY

Der Operator „Cointains All“ steht, wie der Operator „Contains Any“, auch ohne interMedia Text zur Verfügung. Bei Eingabe mehrerer Begriffe entspricht „CONTAINS ALL“ dem booleschen UND–Operator. Es werden die Dokumente ausgegeben, bei denen alle Suchwörter auftreten.
„CONTAINS ANY“ gibt die Ergebnisse aus, bei denen mindestens eins der gesuchten Wörter auftritt, entspricht also einem booleschen ODER–Operator.
Diese Operatoren können auch mit der Soundex und der Fuzzy–Suche kombiniert werden.

7.6.3 Soundex

Soundex[43] ist ein phonetisches Verfahren in dem Wörter, die ähnlich klingen aber unterschiedlich geschrieben werden, bei der Ausgabe der Suchergebnisse mit berücksichtigt werden.
Dieses Verfahren ist für die englische Sprache implementiert und kann, wenn auch weniger effektiv für andere Sprachen eingesetzt werden.
Mit diesem Verfahren ist es möglich, dass z.B. durch die Eingabe des Begriffes *Meyer* auch nach dem Begriff *Meier* gesucht wird.
Soundex reduziert jedes Wort auf einen eindeutigen maximal vier Zeichen (ein Buchstabe und drei Zahlen) langen Code[44] so dass die Suche über einen invertierternIndex erfolgen kann. Aufgrund seiner Einfachkeit ist der Soundex–Algorithmus aber nicht besonders effektiv und wird hauptsächlich für die Suche von Eigennamen verwendet.

7.6.4 Stem

Die Suchbegriffe werden beim Stemming so erweitert, dass auch nach Wörtern mit dem gleichen Stamm gesucht wird. Dies geschieht automatisch und muß bei der Sucheingabe nicht weiter spezifizert werden.

[43] Vgl. [Soundex]
[44] Vgl. [NARA]

7.6.5 Fuzzy

Bei der Fuzzy-Suche werden die Suchwörter aproximiert. Dabei werden die Eingaben nach typischen Buchstabenverdrehern oder fehlerhaften Pre- oder Suffixen untersucht[45].

Der Fuzzy Operator kann hilfreich sein, wenn die korrekte Schreibweise eines Wortes nicht bekannt ist oder in den Dokumenten oft falsch geschrieben auftritt. Er kann aber auch zu einer viel zu großen Menge an Ergebnissen mit einer geringen Trefferquote führen.

Die Gewichtung und der Änhlichkeitsgrad der Suchergebnisse können nicht direkt über die Portalsuche gesteuert werden.

Unter SQL*Plus kann die Abfrage mit Parameter erfolgen:

```
FUZZY(term [, fuzzy_score [, fuzzy_numresults [, weight]]])
```

Über die Variable FUZZY_SCORE kann der Abweichungsgrad definiert werden, wobei ein Wert 80 für identisch steht und bis 50 geändert werden kann. FUZZY_NUMRESULTS steuert die maximale Anzahl ausgegebener Ergebnisse und WEIGHT stellt ein Gewichtungsparameter dar, der über die Auftrittshäufigkeit der Wörter ein besseres Ergebniss zu erzeugen versucht.

[45] vgl [Girill, 1996]

8 Abschluss

8.1 Zusammenfassung

Portal ist ein sehr mächtiges aber auch komplexes Werkzeug um Websites und Intranetdokumente zusammen mit Unternehmensdaten zu verwalten.

8.2 Fazit

Ein nicht zu unterschätzdender Vorteil des Systems ist, dass es komplett aus einer Hand geboten wird. Zum einen sind die einzelnen Komponenten dadurch aufeinander abgestimmt, zum anderen wird die technische Implementierung vereinfacht. Durch die Integrierung des Portals in die Datenbank profitieren auch die Intermedia-Abfragen von den Optimierungsprozessen von Oracle.

Dass sich das Portal bei einer schon vorhandenen Oracle-Installation ohne weitere Lizenzkosten benutzen lässt, dürfte sich aus kostentechnischer Sicht als Vorteil erweisen. Trotzdem ist die Oracle-Onlinedokumentation aufgrund der komplexität des Produktes sehr zerstückelt und beinhaltet eine große Anzahl toter Links. Auch die Installation erweist sich als hackelig. Sind diese Hürden überwunden, erweist sich das System aber zumindest subjektiv als stabil.

Um alle Vorteile der Software zu nutzen, müsste eine Website für das Portal konzipiert und innerhalb des Portals implementiert werden. Um alte Inhalte während einer Migrationszeit zur Verfügung zu stellen, oder aber auch um externe Inhalte in eine eigene Suchmaschine zu integrieren könnte sich die hier vorgestellte Methode als nützlich erweisen.

Während des Schreibens dieser Arbeit, sind schon Oracle 9iAS-Release 2 Versionen für Unix-Systeme erschienen. Laut Beschreibung arbeitet diese Version mit die Oracle 9i-Datenbank als Backend zusammen und bietet bereits von Hause aus ein Werkzeug um Websites zu durchcrawlen (UltraSearch).

8.3 Ausblick

Es besteht keine Frage über die Notwendigkeit Datenbestände effizient durchsuchen zu können. Im Rahmen dieser Notwendigkeit entstehen neue Produkte, die ihre Funktionalität gezielt in dieser Richtung erweitern. Eins dieser Produkte ist „Oracle9i InterMedia Text“. Im Vergleich zur Vorgängerversion wurde diese Funktionalität sogar in die eigentliche Datenbank integriert, was von der Wichtigkeit dieser Entwicklung zeugt.

Glossar

URL– und Literaturverzeichnis

[Alonso, 2001] Omar Alonso: “Oracle Text – An Oracle Technical White Paper, May.2001“, Oracle Corporation 2001, <http://technet.oracle.com/products/text/pdf/text_techwp.pdf> (17.10.2002)

[El-Mallah, 2002] Mohamed El-Mallah: „Web Development with Oracle Portal“ New Jersey: Prentice Hall PTR, 2002

[Faeskorn, 2000] Prof.Dr. H. Faeskorn-Woyke: „Datenbanken und Informationssysteme LE 000615“, ifV NRW, 2000

[Flanagan, 1996] David Flanagan: “Java in a Nutshell” Dt. Ausgabe - 1. Aufl. Bonn: O’Reilly Verl., 1996

[Girill, 1996] T. R. Girill und Clement H. Luk: “Fuzzy Matching as a Retrieval-Enabling Technique for Digital Libraries”, 1996 <http://www.asis.org/midyear-96/girillpaper.html>(12.10.2002)

[Kaminaga] Garrett Kaminaga: “Oracle8i interMedia Text 8.1.7 - Technical Overview “ , <http://otn.oracle.com/products/text/x/Tech_Overviews/imt_817.html> (19.10.2002)

[Koster, 1994] Martijn Koster: „A Standard for Robot Exclusion”, 1994 <http://www.robotstxt.org/wc/norobots.html> (12.10.2002)

[Metalink 130328.1] Metalink DOC ID: Note 130328.1 “How to add more Languages to Oracle Portal” <http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_id=130328.1&p_database_id=NOT> (17.10.2002)

[Metalink 132689.1] Metalink DOC ID: Note 132689.1 “Strategy for maintaining an Intermedia Text Index (using CTX_SCHEDULE)”, 17.1 2001 <http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=132689.1> (19.10.2002)

[Metalink 132693.1] Metalink DOC ID: Note 132693.1 “CTX_SCHEDULE - Utility for maintaining an InterMedia Text Index” <http://metalink.oracle.com/metalink/plsql/ml2_documents.showDocument?p_database_id=NOT&p_id=132693.1> (19.10.2002)

[METALINK] Oracle Corporation: „Oracle Metalink“ , 2002 <http://metalink.oracle.com>(12.10.2002)

[Muench, 2000] Steve Muench: “Building Oracle XML Applications”, 1. Auflage, O’Reilly Verlag, Okt. 2000

[Münz, 2001] Stefan Münz: “SELFHTML: Version 8.0” , 27.10.2001. <http://www.netzwelt.com/selfhtml/navigation/html.htm#verweise> <http://www.netzwelt.com/selfhtml/navigation/html.htm#frames> (12.10.2002)

[NARA] The National Archives and Records Administration: “Soundex Indexing”, College Park, MD 20740-6001, < http://www.archives.gov/research_room/genealogy/census/soundex.html> (18.10.2002)

[OTN A77063-01] Oracle Corporation: „OTN : Oracle8i interMedia Text Reference Release 2 (8.1.6) Part Number A77063-01” <http://otn.oracle.com/docs/products/oracle8i/doc_library/817_doc/inter.817/a77063/cddlpk13.htm#1697> (19.10.2002)

[OTN: SQL*Plus] Oracle Corporation: „OTN : SQL*Plus“ <http://otn.oracle.com/tech/sql_plus/content.html> (19.10.2002)

[OTN] Oracle Corporation: “Oracle Technology Network”, <http://otn.oracle.com> (17.10.2002)

[Pepper,2001] Jason Pepper: “Oracle9iAS Portal 3.0.9.8.2 Architecture & Scalability Overview - An Oracle White Paper”, 2002, <http://portalstudio.oracle.com/pls/ops/docs/FOLDER/COMMUNITY/OTN_CONTENT/MAINPAGE/ARCHITECTURE/30982_ARCH_SCALE_OVERVIEW.PDF> (19.10.2002)

[Portal A86707-01] Oracle Corporation: “Oracle Portal 3.0 Configuration Guide Release 3.0 Part Number A86707-01 Using intermedia Text in Oracle Portal” <http://rainbow.mimuw.edu.pl/oracle9iAS/portal.102/a86707/chapter5.htm> (19.10.2002)

[Portal A90096-01] Oracle Corporation: “Oracle9iAS Portal Configuration Guide Release 3.0.9 Part Number A90096-01 Setting up the Search Feature in Oracle Portal Content Areas” <http://otn.oracle.com/docs/products/ias/doc_library/1022doc_otn/portals.102/a90096/cgsearch.htm> (19.10.2002)

[Soundex] Oracle Corporation: “Oracle Text Reference Release 9.0.1Part Number A90121-01 - Contains Query Operators,18 of 28” < http://download-west.oracle.com/otndoc/oracle9i/901_doc/text.901/a90121/cqoper18.htm#14551> (19.10.2002)

[Technet : add_content_area] Oracle Corporation: “Technet PLSQL documentation: add_content_area function“ <http://technet.oracle.com/products/iportal/files/pdk/plsql/doc/sdk23aca.htm> (19.10.2002)

[Technet : add_item] Oracle Corporation: “Technet PLSQL documentation: add_item function“ <http://technet.oracle.com/products/iportal/files/pdk/plsql/doc/sdk23ai.htm> (19.10.2002)

[Technet: add_folder] Oracle Corporation: “Technet PLSQL documentation: add_folder function“ <http://technet.oracle.com/products/iportal/files/pdk/plsql/doc/sdk23af.htm> (19.10.2002)

[Technet: API constants] Oracle Corporation: “Technet PLSQL documentation: Content Area API constants“ <http://technet.oracle.com/products/iportal/files/pdk/plsql/doc/sdk23con.htm> (19.10.2002)

[Technet: API packages] Oracle Corporation: “Technet PLSQL documentation: Content area API packages“ <http://technet.oracle.com/products/iportal/files/pdk/plsql/doc/sdk23pkg.htm#rel> (19.10.2002)

[Technet: Content Area Views] Oracle Corporation: “Technet PLSQL documentation: Secured content area views“ <http://technet.oracle.com/products/iportal/files/pdk/plsql/doc/sdk23vws.htm> (19.10.2002)

[Technet: submit_search] Oracle Corporation: “Technet PLSQL documentation: submit_search procedure“ <http://technet.oracle.com/products/iportal/files/pdk/plsql/doc/sdk23ss.htm> (19.10.2002)

[UltraSearch] Oracle Corporation: “Oracle Ultra Search 9.0.1.0.0 README”, 2001, <http://technet.oracle.com/products/ultrasearch/htdocs/9.0.1README.html> (19.10.2002)

[UNICODE] Oracle Corporation: „Globalization Support: Oracle Unicode database support An Oracle White Paper”, Februar 2002, <http://otn.oracle.com/tech/globalization/pdf/Unicode.PDF> (18.10.2002)

[Vanhelsuwé, 1996] Laurence Vanhelsuwé: “Automating Web Exploration”, 1996, <http://www.javaworld.com/javaworld/jw-11-1996/jw-11-webcrawler.html>(14.10.2002)

Erklärung

Ich versichere, die von mir vorgelegte Arbeit selbständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt, bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Bonn, den ..... Unterschrift:.....